2000-09

Integrating realistic human group behaviors into a networked 3D virtual environment

Miller, Thomas Erik

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

INTEGRATING REALISTIC
HUMAN GROUP BEHAVIORS
INTO A
NETWORKED 3D VIRTUAL ENVIRONMENT

by

Thomas Erik Miller

September 2000

Thesis Advisor: Donald P. Brutzman
Second Reader: Patrick V. Mack

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 2000	Master's Thesis

4. TITLE AND SUBTITLE
Integrating Realistic Human Group Behaviors Into A Networked 3D Virtual Environment.

5. FUNDING NUMBERS

6. AUTHOR(S)
Thomas Erik Miller

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT

Virtual humans operating inside large-scale virtual environments (VE) are typically controlled as single entities. Coordination of group activity and movement is usually the responsibility of their "real world" human controllers. Georeferencing coordinate systems, single-precision versus double-precision number representation and network delay requirements make group operations difficult. Mounting multiple humans inside shared or single vehicles, (i.e. air-assault operations, mechanized infantry operations, or small boat/riverine operations) with high fidelity is often impossible.

The approach taken in this thesis is to reengineer the DIS-Java-VRML Capture the Flag game geolocated at Fort Irwin, California to allow the inclusion of human entities. Human operators are given the capability of aggregating or mounting nonhuman entities for coordinated actions. Additionally, rapid content creation of human entities is addressed through the development of a native tag set for the Humanoid Animation (H-Anim) 1.1 Specification in Extensible 3D (X3D). Conventions are demonstrated for integrating the DIS-Java-VRML and H-Anim draft standards using either VRML97 or X3D encodings.

The result of this work is an interface to aggregate and control articulated humans using an existing model with a standardized motion library in a networked virtual environment. Virtual human avatars can be mounted and unmounted from aggregation entities. Simple demonstration examples show coordinated tactical maneuver among multiple humans with and without vehicles. Live 3D visualization of animated humanoids on realistic terrain is then portrayed inside freely available web browsers.

14. SUBJECT TERMS
Virtual Environments, Humanoid Animation 1.1 Specification, Distributed Interactive Simulation, 3D, aggregation, mounting human entities, virtual humans, avatars, X3D, X3d-Edit, VRML, Java, DIS-Java-VRML, Web3D Consortium

15. NUMBER OF PAGES
136

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

INTEGRATING REALISTIC HUMAN GROUP BEHAVIORS INTO A NETWORKED 3D VIRTUAL ENVIRONMENT

Thomas Erik Miller
Major, United States Army
B.S., University of Virginia, 1989

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MODELING, VIRTUAL ENVIRONMENTS, AND SIMULATION

from the

NAVAL POSTGRADUATE SCHOOL
September 2000

Author: _____

Thomas Erik Miller

Approved by: _____

Donald P. Brutzman, Thesis Advisor

_____

Patrick V. Mack, Second Reader

_____

Rudy Darken, Academic Associate
Modeling, Virtual Environments, and Simulation Academic Group

_____

Michael Zyda, Chair
Modeling, Virtual Environments, and Simulation Academic Group

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Virtual humans operating inside large-scale virtual environments (VE) are typically controlled as single entities. Coordination of group activity and movement is usually the responsibility of their "real world" human controllers. Georeferencing coordinate systems, single-precision versus double-precision number representation and network delay requirements make group operations difficult. Mounting multiple humans inside shared or single vehicles, (i.e. air-assault operations, mechanized infantry operations, or small boat/riverine operations) with high fidelity is often impossible.

The approach taken in this thesis is to reengineer the DIS-Java-VRML Capture the Flag game geolocated at Fort Irwin, California to allow the inclusion of human entities. Human operators are given the capability of aggregating or mounting nonhuman entities for coordinated actions. Additionally, rapid content creation of human entities is addressed through the development of a native tag set for the Humanoid Animation (H-Anim) 1.1 Specification in Extensible 3D (X3D). Conventions are demonstrated for integrating the DIS-Java-VRML and H-Anim draft standards using either VRML97 or X3D encodings.

The result of this work is an interface to aggregate and control articulated humans using an existing model with a standardized motion library in a networked virtual environment. Virtual human avatars can be mounted and unmounted from aggregation entities. Simple demonstration examples show coordinated tactical maneuver among multiple humans with and without vehicles. Live 3D visualization of animated humanoids on realistic terrain is then portrayed inside freely available web browsers.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF SYMBOLS, ACRONYMS AND ABBREVIATIONS

Π	Pi, approximately 3.14
±	Plus or minus
*.wrl	World file, file extension for a VRML file
*.wrz	Compressed World (VRML) file
2D	Two Dimensions, Two-Dimensional
3D	Three Dimensions, Three-Dimensional
AAR	After-Action Review
BOS	Battlefield operating Systems
COA	Course of Action
CS	Computer Science, The Department of Computer Science
CTF	Capture the Flag (game)
DAG	Directed Acyclic Graph
DC VET	Damage Control Virtual Environment Trainer
DI	Dismounted Infantry
DIS	Distributed Interactive Simulation (protocol)
DJV	DIS-Java-VRML
DoD	Department of Defense
DOM	Document Object Model
DST	Decision-Support Template
DTD	Document Type Definition
ECMAScript	JavaScript
ESPDU	Entity State Protocol Data Unit
EXTERNPROTO	VRML External Prototype
FAQ	Frequently Asked Questions
fo	Formatting Object supported by XSL
FOG-M	Fiber-Optically Guided Missile
FOM	HLA Federation Object Model
GUI	Graphical User interface
H-Anim	Human Animation
H-Anim 1.1 spec	Human Animation 1.1 Specification
HLA	High Level Architecture
HTML	Hyper Text Markup Language
IEEE	The Institute of Electrical and Electronic Engineers
IP	Internet Protocol
IPB	Intelligence Preparation of the Battlefield
ISO	International Standards Organization
Java3D	An extension of Java for 3D graphics
LAN	Local-Area Network
LSVE	Large-Scale Virtual Environments
MOUT	Military Operations in Urban Terrain

MOVES	Modeling, Virtual Environments and Simulation
NPS	Naval Postgraduate School
NPSNET	A family of LSVEs developed at NPS (NPS Networked Vehicle Simulator)
NTC	National Training Center
PDU	Protocol Data Unit
PROTO	VRML Prototype
SGI	Silicon Graphics Incorporated
SGML	Standard Generalized Markup Language
SIMNET	Simulator Networking, Simulation Network
SOF	Special Operations Forces
SF	Special Forces
STRICOM	US Army Simulation, Training and Instrumentation Command
TEWT	Tactical Exercise Without Troops
UPenn	University of Pennsylvania
URL	Uniform Resource Locater
US	United States, United States of America
VE	Virtual Environment
VRML	Virtual Reality Modeling Language
VRTP	Virtual Reality Transfer Protocol
W3C	World Wide Web Consortium
WEB3D	Web 3D Consortium formerly known as the VRML Consortium
X3D	Extensible 3D, Extensible 3D Graphics specification
Xj3D	X3D/VRML-Java3D – open-source VRML/X3D web browser written in Java3D
XML	Extensible Markup Language
XSL	Extensible Stylesheet Language

ACKNOWLEDGEMENTS

I would like to acknowledge the body of open-source distribution work from the NPSNET Research Group, the MOVES Academic Group and Faculty, and the Web3D Consortium, in particular, the X3D Specification Task Group and the DIS-Java-VRML and Human Animation Working Groups, that made this thesis research possible.

I would like to thank Cindy Ballreich for allowing the use of her model Nancy. I would also like to thank my advisors, Dr. Don Brutzman and LT Pat Mack USN, for their technical expertise, support, focus, guidance and friendship throughout this project.

Last but not least, I would like to thank my family (Susan, Isabelle and Traveller) for their unending patience and perseverance. Truly, without the loving support of my wife Susan, this work could not have been completed.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

To
The Inhabitants of SPACE IN GENERAL
And H.C. IN PARTICULAR
This Work is Dedicated
By a Humble Native of Flatland
In the Hope that
Even as he was Initiated into the Mysteries
Of THREE Dimensions
Having been previously conversant
With ONLY TWO
So the Citizens of that Celestial Region
May aspire yet higher and higher
To the Secrets of FOUR FIVE OR EVEN SIX Dimensions
Thereby contributing
To The Enlargement of THE IMAGINATION ….
– Dedication from Flatland: A Romance of Many Dimensions
by A. Square (Abbott, 1992)

A. BACKGROUND

Visualization of a battle space in three dimensions (3D) is a traditional training task for combat arms leaders of the United States Army. The Army tries to develop this skill through formalized professional training in courses, exercises such as Staff Rides to historical battle sites, terrain walks, and Tactical Exercises without Troops (TEWTs), as well as through the use of briefing techniques such as sand tables and operational overlays. Even though not yet in widespread use, this skill can be further developed through simulations in virtual environments.

The Department of Defense (DoD) is the largest developer of networked virtual environments, having developed SIMNET and Distributed Interactive Simulation (DIS),

and was also an early explorer in large-scale virtual environments (LSVEs) (Singhal, 1999). DoD is committed to the use of training in virtual environments to improve the quality of its "warriors" more effectively and economically. However, in high-resolution simulations, the focus has been on vehicular systems being exercised as individual entities. The use of human entities or dismounted infantry (DI) has been limited in such virtual simulations (Reece, 2000).

This thesis is an effort to add human entities into the existing virtual environment of DIS-Java-VRML's canonical game example, Capture the Flag (CTF). These entities are capable of being controlled as an aggregate unit (fire team) or individually. They can also aggregate or mount, move with and move within a non-human entity (in this case, a helicopter). Such mounting enables precise positioning and control of related entities even when operating distant, distributed controllers over georeferenced terrain.

B. MOTIVATION

The motivation for this project is to provide higher headquarters, mission planners and mission executors with a visualization tool that can realistically display a mission's area of operations. Additionally, the design of this project facilitates the sharing of ideas, information and operational concepts based on the visualization even if headquarters, planners and executors are not geographically collocated. Three possible uses are as a virtual sand table, a networked virtual rehearsal tool and a tool for after-action review (AAR). Specific details regarding these scenarios are now examined.

1. 3D Mission Visualization – Virtual Sand Table

Actual terrain models, commonly referred to as sand tables, are detailed, scaled-3D representations of a geographical region representing an area of operations. Sand tables typically show accurate or exaggerated elevation and relief. Sand tables are usually large (between one and ten meters squared) and are often established in a central location within the planning area to allow multiple planners to simultaneously access it and to as an area to conduct staff coordination. Such models enable the inclusion of mission graphics to include routes, boundaries, targets and objective areas, as well as troop locations and decision-support templating. Sand tables provide excellent understanding and knowledge of the terrain and also provide a means for a unit to visualize a complete mission from start to finish.

Ideally, a virtual sand table may provide rapid generation of a geolocated 3D model for any location in the world. Subsequently, it also needs to allow the inclusion of mission graphics and troop locations. It then can be used in support of the intelligence preparation of the battlefield (IPB) process and decision process, used by the US Army and Marine Corps, in such areas as course of action (COA) development and COA selection, war-gaming, and the creation of synchronization matrices and decision-support templates. The IPB process, synchronization matrices and decision-support templates are discussed in Chapter II.

2. Networked Virtual Rehearsal

Mission executors, particularly Special Operation Forces (SOF), are commonly geographically disparate from their command elements, final mission approval authority,

and even support elements such as aviation or sea borne transport. A tool that allows all participants to rehearse a mission from their current, disjoint locations creates greater understanding of overall mission roles, and might also be done in a manner more secure and requiring less bandwidth then current procedures. In terms of mission approval, such a capability can allow the decision maker to see a 3D execution of the mission rather than solely a written plan of execution with supporting 2D graphics. It further allows supporting forces greater understanding of the supported forces planned actions in cases of emergencies, mission changes during execution or contingencies.

3. Networked After-Action Review (AAR) or Debriefing Tool

After-actions reviews (AAR) are a critical element in the success and improvement of military readiness and training. Unfortunately, AARs are often constrained by a lack of resources in data recording and visual presentation of spatial relations during the reviewed events. Debriefings are often troublesome because of an inability to appropriately describe or understand complex spatial data regarding a mission setting. A tool that provides the ability to display spatial relationships in a 3D representation of the actual mission or training location can certainly augment current practice to provide a more thorough understanding and better learning experience. Additionally, for debriefings, playback of recorded behaviors allows a more complete and accurate report. Addition of "what if?" scenario alternatives might provide significant new planning and training capabilities.

C. OBJECTIVES

The objective of this research is to provide an interface to aggregate and control articulated humans in a networked virtual environment. To achieve this objective, the following areas are addressed:

- Virtual human avatars must have an articulated joint structure and at least a limited motion library in order to model realistic movement.

- A set of rule-based physical and logical behaviors for groups of humans must be developed and implemented in order to execute basic tactical formations and activities.

- Human entities must be able to aggregate into a group or mount other non-human entities (such as vehicles) and then separate back to individual entity control. Otherwise, the high-precision relative motion needed for group activities is not possible across network delays or in georeferenced locations.

- The source code for this research ought to be platform-independent and open-source distribution. Ideally, the code ought to be deployable (in the military sense) worldwide with a military unit regardless of their computer systems, operating efficiently if not optimally.

- The final product must be "visually compelling" to make it a viable visualization tool. The system must have enough appeal, usability and practical value to encourage user acceptance.

D. THESIS OUTLINE

This chapter describes the background, motivation and objectives for integrating human group behaviors into the DIS-Java-VRML CTF game. Chapter II provides background of DIS-Java-VRML and the CTF game, humanoid animation, the Virtual

Reality Modeling Language (VRML) and Extensible 3D (X3D), and NPSNET IV work with human animation. Chapter III provides a clear and concise problem statement for this thesis. Chapter IV discusses initial development efforts to include integrating humans into the CTF game, CTF usability improvements and an examination of VRML browsers. Chapter V describes the inclusion of the H-Anim 1.1 specification into X3D-Edit editing tool and the creation of H-Anim 1.1 specification compliant humanoids using this tool. Chapter VI discusses the mounting of the human entities in the virtual environment. Chapter VII describes the software providing the multi-entity, single-user interface, the group-control panel and the group behaviors. Chapter VIII provides thesis conclusions and recommendations for follow-on or future research.

II. RELATED WORK

A. INTRODUCTION

This chapter provides an overview to DIS-Java-VRML framework and the humanoid models used in this research. Further, it examines pertinent background work in large-scale virtual environment (LSVE) research and in the animation of virtual humans on the World Wide Web or other networks. It also provides an introduction to the new and developing technologies used in support of this thesis, and a brief overview of the US Army's IPB and decision processes.

B. DIS-JAVA-VRML

> A way to implement networked, shared virtual worlds using widely available technology and tools. (McGregor, 2000).

In the development of the Virtual Reality Transfer Protocol (VRTP), the marriage of the IEEE Standard Distributed Interactive Simulation (DIS) protocol for networking, the flexible object-oriented programming language Java, and the Virtual Reality Modeling Language (VRML) for 3D graphics has taken the process a fundamental step forward. The work follows in the steps of NPSNET IV by using DIS and Internet Protocol (IP) multicast. The work greatly expands the group of possible users by making use of widely available technologies, platform independence, open-source software code and open standards. DIS-Java-VRML is a Web3D Consortium working group that works in parallel with the VRTP working group.

1. Distributed Interactive Simulation (DIS)

> Distributed Interactive Simulation (DIS) is a government/industry initiative to define an infrastructure for linking simulations of various types at multiple locations to create realistic, complex virtual 'worlds' for the simulation of highly interactive activities (IEEE, 1995).

The Distributed Interactive Simulation (DIS) protocol describes an IEEE approved standard of communications between entities in distributed simulations (IEEE, 1993). The standardization of object interactions and large lexicon of behavior-based messages provides an excellent interface for general usage in LSVEs.

The Protocol Data Unit (PDU) is the structure used to pass information between entities and from the simulation to an entity and back. Twenty seven PDU types are defined in the 1995 IEEE Standard for DIS-Application Protocols (IEEE, 1995) as shown in Table 1, which fall into seven categories (Entity Information/Interaction, Warfare, Logistics, Simulation Management, Distributed Emission Regeneration and Radio Communications). The Entity State PDU (ESPDU) is the principal message type for DIS transmissions because it carries the physical (location and motion) state information of an entity. To maximize positional information transfer while reducing bandwidth, standardized dead reckoning algorithms are available. Even though 27 PDUs exist, in most DIS-compliant VEs only three to four of the most commonly used PDUs (ESPDU, Fire, Detonation and Collision) are used. Free and commercial DIS libraries are widely available, including C++ and Java implementations from NPS.

Protocol Data Unit	Purpose
Entity State *	Update entity state, i.e. location, rotation, velocity, acceleration etc.
Fire *	Reports firing of a weapon
Detonation *	Reports the impact or detonation of a munition
Collision *	Reports the collision of an entity with an object or other entity
Service request	Request for the transfer of supplies
Resupply Offer	Offers supplies to another entity
Resupply Received	Acknowledges the receipt of some or all offered supplies
Resupply Cancel	Cancels a resupply request
Repair Complete	Report Completion of a repair
Repair Response	Acknowledges the completion of a repair
Start/Resume *	Instructs an entity to participate in a simulation
Stop/Freeze *	Instructs an entity to stop or pause its participation in a simulation
Acknowledge *	Acknowledges the receipt of a Start, Stop, Create or Remove PDU as part of simulation management
Action Request *	Requests a specific action from an entity
Action Response *	Acknowledges the receipt of an Action Request
Data Query *	Requests data from another entity
Set Data *	Sets or changes the parameters in an entity
Data *	Returns data requested by a Data Query or Set Data PDU
Event Report *	Communicates the occurrence of a significant event such as entry into a minefield
Message	Sends a miscellaneous unformatted message
Create Entity *	Establishes the identity of a new entity
Remove Entity *	Removes an entity from the simulation
Receiver *	Communicates the state of a radio receiver
Signal *	Transmit voice or data
Transmitter *	Announce start/stop of radio transmissions
Designator	Communicates designation emissions
Electromagnetic Emission	Communicate electromagnetic emissions other than radios

Table 1. Distributed Interactive Simulation Protocol Data Units. (Note: * implemented in DIS-Java-VRML). From (Stewart, 1996).

2. Java

Java is a programming language designed by the engineers at Sun Microsystems starting in 1991. In a white paper written by the designers, released in 1995 before Java's official release in January 1996, they gave 11 key words that explained their design goals (Horstmann, 1999). They are Simple, Object Oriented, Distributed, Robust, Secure, Architecture Neutral, Portable, Interpreted, High Performance, Multithreaded and Dynamic. Widespread Java use was propagated when Netscape, in the fall of 1995, decided to make Netscape browsers (starting with Netscape 2.0) Java enabled. Sun has continued to release upgrades to Java releasing Java 1.2 or Java 2 in December of 1998. Development and release of Java 1.3 has already occurred for some platforms (notably Windows, Linux and Solaris) but is still in beta testing for others. (Horstmann, 1999).

Java, through its design conception and improvements, offers a number of desirable attributes for its use as an application in a LSVE. It is cross platform, meaning multiple hardware architectures and operating systems, providing portable code that does not require recompilation. It offers network support and can operate with the Netscape and Microsoft security models. Of key interest to this research: Java is VRML compatible, with three class libraries specified for the Java-VRML interface. Finally, if not yet high performance as visualized by its designers, it already provides excellent performance with continuing steady improvement. (Brutzman, 2000).

3. Virtual Reality Modeling Language (VRML)

The Virtual Reality Modeling Language (VRML) is a general 3D scene description language for describing interactive 3D objects and worlds, and intended to be

a universal interchange format for integrated 3D graphics and multimedia. It is capable of representing static and animated dynamic 3D and multimedia objects with hyperlinks to other media such as text, sounds, movies, and images. It is an International Standards Organization (ISO) specification (ISO/IEC 14772-1:1997). VRML provides a large number of 3D graphics nodes, and they are organized in a hierarchical manner, within a file, to compose a directed acyclic graph (DAG) called a scene graph (Brutzman, 1998). VRML files are called Worlds or scenes and end with a .wrl file name extension (or .wrz for a gzip-compressed VRML file). A VRML file, like the example in Figure 1, can contain four main types of components: the VRML header; Prototypes; Shapes (geometry and appearance), Interpolators, Sensors, and Scripts; and Routes (Ames, 1997). The only element that it must contain is the VRML header. Prototypes (PROTOs) allow a user to author new 3D graphic node types, and can be formed into libraries and reused by referencing them by using an external prototype (EXTERNPROTO) declaration. Shapes encompass both object geometry and appearance. VRML geometries include 3D geometry primitives, text, and face sets, and appearances include color components, transparency and textures. Interpolators define keys for key frame animations. Sensors allow users to interact with the scene. Script nodes are shells which provide which provide an interface between the VRML scene and a program script, usually written in Java or JavaScript. They are crucial in creating complex actions and animations to include networked game play and articulated motion. Finally, routes are statements which define connections (or routes in the sense of an electric circuit) between named nodes and fields, which allow events to be passed from source to target.

Figure 1. VRML Scene and Source hello-world.wrl. From (Brutzman, 1998).

4. Capture The Flag (CTF) Game

The Capture the Flag (CTF) game is a demonstration of an interactive 3D virtual environment implemented through DIS-Java-VRML and is part of the DIS-Java-VRML source code at *http://www.web3d.org/WorkingGroups/vrtp/dis-java-vrml*. It uses IP Multicast and is implemented in a server-less peer-to-peer fashion. It was largely written by students at NPS over a two-year period as a series of class projects for CS4472, Advanced Physically Based Modeling. The basic scenario is two teams, red and blue, competing to capture their opponents' flag and safely return with it to their home base,

which is an airfield. The game is geolocated at the National Training Center (NTC) at Fort Irwin, California. The red flag and red airfield is shown below in Figure 2. Each team originally consisted of three M1 Abrams tanks and three OH-58 Kiowa helicopters. The entities are controlled through "control panels" which are Java frames, external to the scene. The frames take user input information, translates it using Java logic in an "Action Interpreter" and updates the scene, by passing protocol data units (PDUs) through a Java script to a VRML PROTO node called the EspduTransform node, as shown in Figure 3.

Figure 2. Screen Shot of a Blue Tank With Trace Values Displayed and the Blue Flag in CTF.

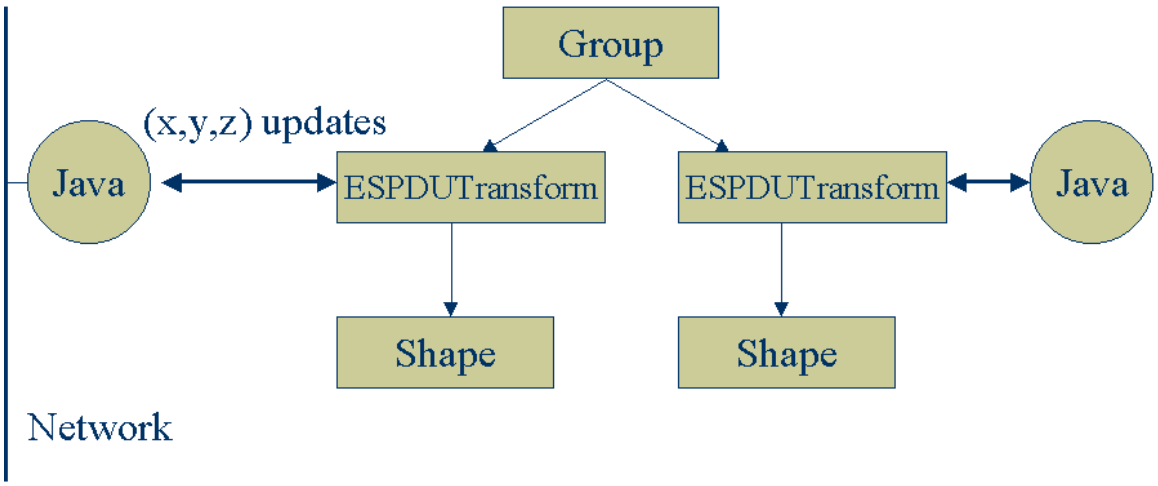


Figure 3. DIS-Java-VRML Flow Chart Showing Multiple Networked Entities in a Single Web Browser. From (McGregor, 2000).

5. Coordinate Systems

The coordinate axes as defined by the VRML and DIS standards are different in orientation. VRML defines its ground plane in terms of the XZ plane with positive Y defining up. DIS defines its ground plane in terms of the XY plane with negative Z defining up. See Figure 4. Both systems define positive X to be North in world coordinates and the nose of the entity in body coordinates. The positive orientation of the other axis defining the ground plane represents East in world coordinates or to the entity's right in body coordinates. The translation between the two coordinate systems is transparent to the operator because it is handled by the ESPDUTransform script. However, properly handling such spatial correspondences is crucially important when designing applications for CTF.

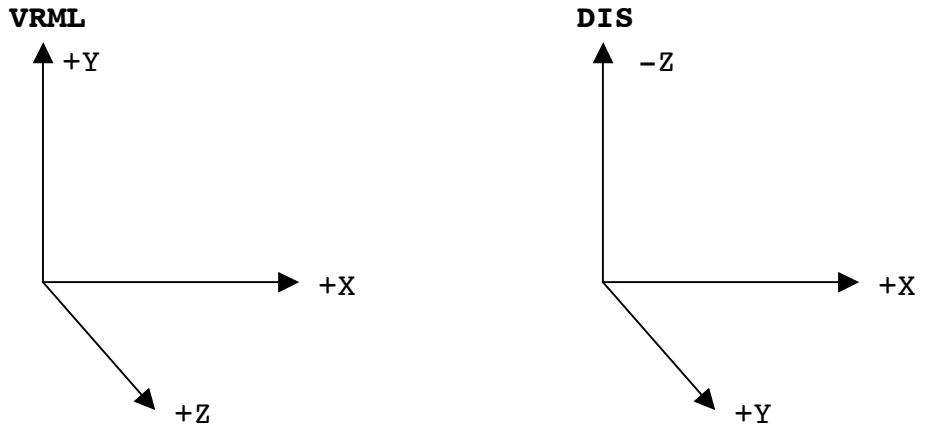


Figure 4. Comparison of the VRML and DIS Coordinates Axes.

C. THE HUMAN ANIMATION 1.1 SPECIFICATION (H-ANIM 1.1 SPEC)

The Human Animation 1.1 Specification (H-Anim 1.1 spec) was developed by Human Animation Working Group of the Web3D Consortium in order to provide a means of interoperability of virtual humans and human behaviors for animations created by different authors using different authoring tools, as shown in Figure 5. The working group followed three design goals in implementing this specification: Compatibility, Flexibility, and Simplicity. Though originally constrained to just any VRML 2.0 compliant browser, the drive for compatibility is now to have H-Anim humans work anywhere. In fact, the H-Anim specification has been represented as a High Level Architecture (HLA) Federation Object Model (FOM) (Shockley, 2000). Flexibility assists in the drive for compatibility, allowing the specification to include information that is required by some applications and not by others. Simplicity is manifested since

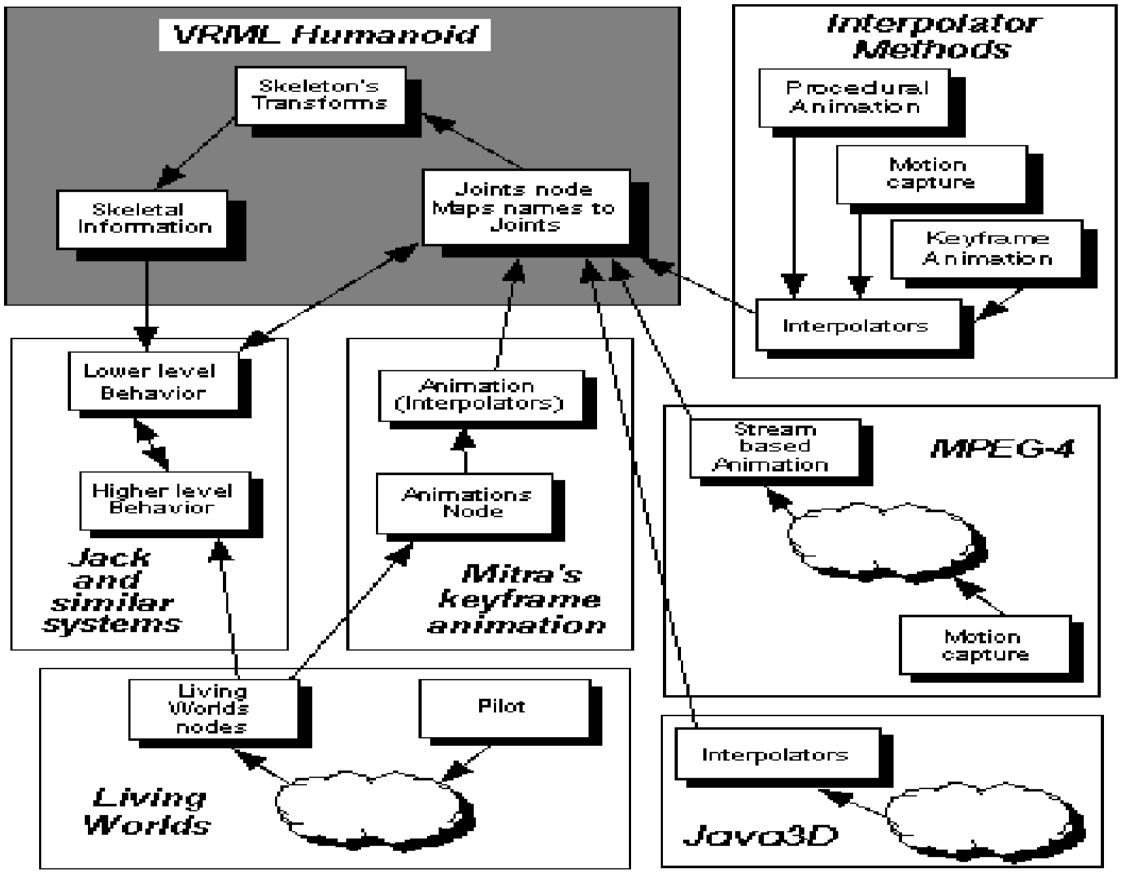


Figure 5. Connection of H-Anim with Other Systems. From (H-Anim, 1999).

the specification only addresses humans and the belief that the specification is not static and can be extended if necessary. (H-Anim, 1999)

The H-Anim specification, as its fundamental basis, defines three VRML PROTO nodes. Segments are the geometries of body parts. Joints are the connectors and controllers of the segments. The Humanoid Proto holds the joints and segments in a hierarchical format as well as metadata and other scene graph information. The most significant contributions of the H-Anim 1.1 spec are the definition of a neutral body pose and the establishment of naming conventions. The neutral body pose defines a position

such that all joint rotations are zero. This is extremely useful for animation as a default position. The naming convention defines the set of names for each joint and segment of the human body as seen in Figure 6. The naming convention includes 94 joints with corresponding segments, and is purported to be sufficient to model a virtual human to the highest level of anatomical correctness. This allows an application to easily access the data concerning a virtual human's geometry. (Roehl, 2000).

D. HUMAN MODELS

A search for a suitable model with a motion library was conducted since the creation of a suitable, realistic, compelling and articulated virtual human avatar with a motion library is difficult, time consuming (Maestri,1999) and fell outside the scope of this thesis research. The model had to be able to be converted to accept the EspduTransform node for network play, and provide access into its motion library so that the motions could be mapped to articulated parameters.

There were a number of available models to choose from. Models from the US Army Simulation, Training, and Instrumentation Command (STRICOM), members of the H-Anim working group (most notably Beitler, Babski, Ballreich, Flavin and Miller) and Blaxxun (among others) were examined. There were four major hindrances among the models with the largest problem being that models that were not 100% H-Anim 1.1 specification compliant. The other major difficulties were lack of adequate, appropriate

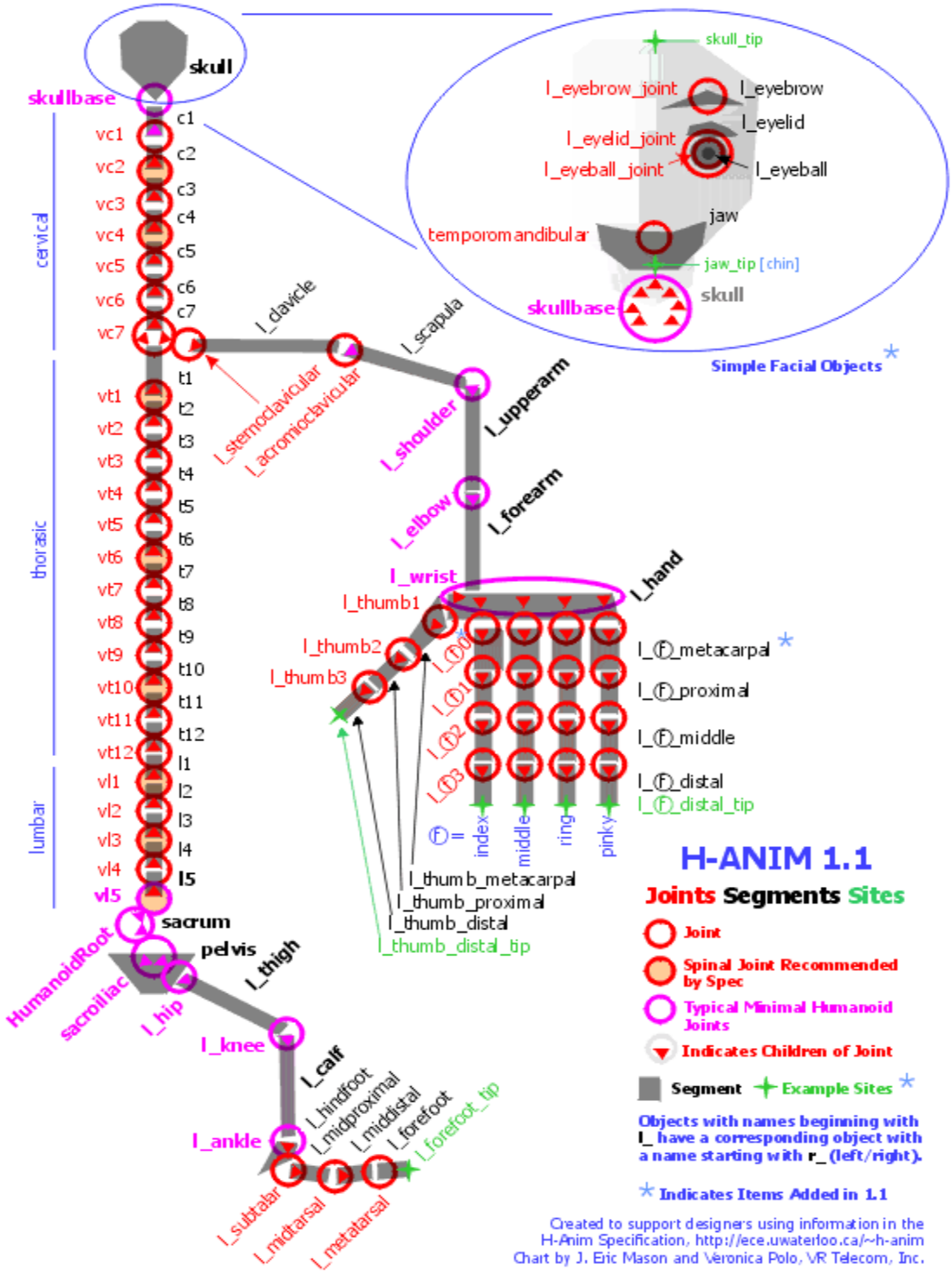


Figure 6. H-ANIM 1.1 Spec Joint and Segment Names with Skeleton Diagram. From (H-Anim, 1999).

motion libraries, models being overly complex and heavyweight (at times in excess of 7,000 polygons) and not being adequately realistic for visual appeal and acceptance (McCloud, 1993 and McCloud, 2000). After examining a number of alternatives, the H-Anim 1.1 compliant Nancy.wrl was chosen as the principal model.

Nancy.wrl is the canonical exemplar of H-Anim 1.1 spec (as it previously was for H-Anim 1.0 spec). The model was created by Cindy Ballreich, who grants permission for its non-commercial usage when distributions include appropriate credits and maintenance of the 3Name3D name and logo. Nancy.wrl consists of 2082 polygons and contains 17 joints, 15 segments and four viewpoints. It also contains a behavior library with four behaviors, stand, walk, run and jump, which are activated by a Touch Sensor. See Figures 7-10.

Figure 7. Nancy, The Canonical H-Anim Exemplar, Demonstrating the Stand Behavior. From (H-Anim, 1999).

Figure 8. Nancy Demonstrating the Walk Behavior. From (H-Anim, 1999).

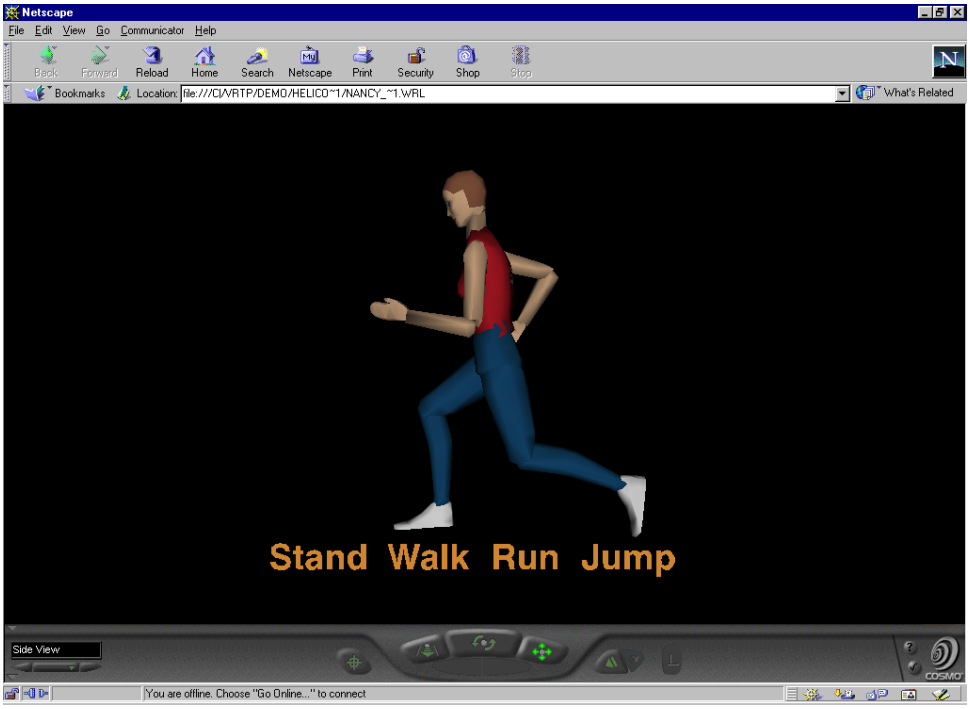


Figure 9. Nancy Demonstrating the Run Behavior. From (H-Anim, 1999).

Figure 10. Nancy Demonstrating the Jump Behavior. From (H-Anim, 1999).

E. NPSNET

NPSNET is a networked virtual environment (VE) for battlefield simulation and was, during its height, the highest-performance VE software available. In 1986, two Computer Science students at the Naval Postgraduate School (NPS) developed a visual simulator for the fiber-optically guided missile (FOG-M) system. The FOG-M, its follow on system, VEH, and the Moving Platform Simulator (MPS) were the precursors to NPSNET. (Singhal, 1999).

NPSNET I was named in March 1990. It played with other systems on a local-area network (LAN), did not use dead reckoning and sent packets at visual frame rate (typically ten packets per second). As seen in Figure 11, parallel development followed with NPSNET II, NPSNET III and then NPSSTEALTH. NPSSTEALTH was operational

in early 1993 and was the only workstation-based VE capable of interoperating with SIMNET. In the fall of 1993, the first version of NPSNET IV became operational. NPSNET-DI supported Dismounted Infantry (DI) men as a special version of NPSNET IV. Of note, research has now begun on NPSNET V with the Virtual Reality Transport Protocol (VRTP) as its means of communication. (Singhal, 1999).

1. Overview of NPSNET IV

NPSNET IV was motivated by the desire to expand VEs beyond LANs and expand VE capabilities using multicast networks to serve more than 1,000 simultaneous users. It ran on commercial, off-the-shelf Silicon Graphics Incorporated (SGI) workstations (at the time the best graphics workstations commercially available) and

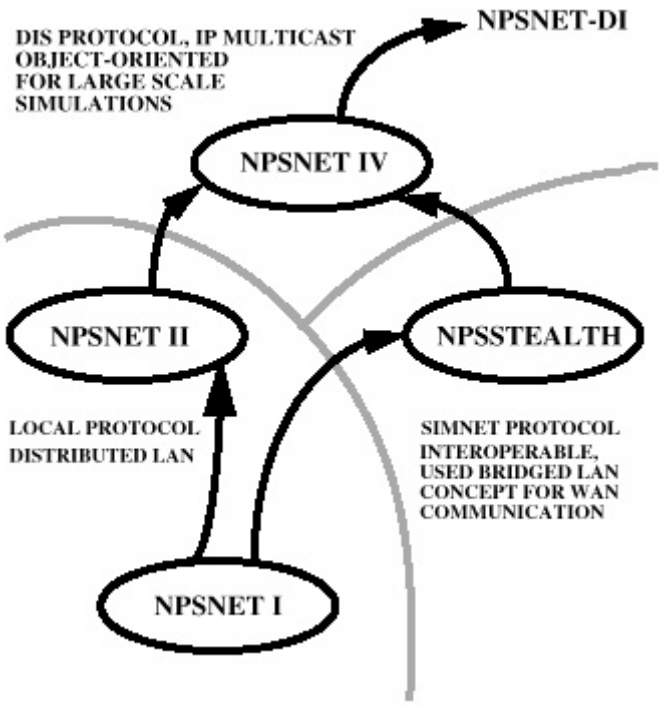


Figure 11. Evolution of NPSNET. From (Macedonia, 1994).

employed the Performer graphics library. NPSNET IV was the first DIS application to use the IP Multicast Protocol. It provided low-cost 3D sound, simulation-based design and autonomous players for populating virtual worlds to include a fully articulated human. (Macedonia, 1995). The NPSNET Research Group planned a major overhaul and redesign of NPSNET IV during the fall of 1993, but such an effort was never undertaken. This software is no longer in widespread use, due to its reliance on (SGI) workstations and heavy administrative support requirements. NPSNET IV continued on through December of 1996, eventually being utilized by over 100 DoD contractors and other university laboratories (Singhal, 1999), as well as a research platform for over 60 NPS students.

2. Human Entities

In 1994, a joint-research project with NPS, SARCOS Inc. and the University of Pennsylvania (UPenn), under Army Research Laboratory sponsorship demonstrated the insertion of a fully articulated human into a DIS environment. The demonstration involved *Jack*, a program to provide and manipulate articulated human geometries from UPENN, NPSNET IV from NPS and Individual Soldier Mobility System (hardware) controller from SARCOS. (Pratt, 1994). While successful, there was significant overhead in communicating to Jack and the SARCOS sensor suit was cumbersome and difficult to adjust.

In 1995, the NPSNET Research group began the NPSNET-HUMAN Project. The ultimate project goal was real-time interaction with entities and objects in a VE with high-resolution feedback. The focus was for tactical medical personnel and dismounted

infantry soldiers. The project achieved success with an accurate articulated human viewpoint, simple networked wounding, a networked autonomous evacuation unit, real-time arm articulations (Waldrop, 1995) and level of detail (LOD) model work allowing 150 DI entities to exist in NPSNET IV (Chrislip, 1995). (Zyda, 1995). However, there was discussion about the its limitations and complexity (Roehl, 1995).

a) Jack

Jack was developed by UPenn's Center for Human Modeling and Simulation headed by Dr. Norman Badler. Jack was originally written as a program to manipulate articulated human geometries (Badler, 1993). Later variations included the Jack Motion library (implemented in NPSNET IV), and JackMOO, a web-based simulation system for virtual humans. In 1996, Transom Corporation (recently acquired by EAI) bought the exclusive rights to develop, commercialize and support Jack software. There have been three commercial releases of Jack, the latest under the trade name TransomJack. See Figure 12.

3. Mounting Human Entities

Three NPS theses address mounting humans in a VE in the NPSNET environment space. In 1995, humans were incorporated into the Damage Control Virtual Environment Trainer (DC VET) by LT James O'Byrne USN (O'Byrne, 1995). However, DC VET was a stand-alone environment and one of the recommendations for follow on research was to incorporate DC VET into NPSNET IV. In 1996, LT Bryan Stewart USN, incorporated DC VET into NPSNET IV and subsequently mounted human entities to a non-human

Figure 12. Three JACK Humans in a Military Setting. From (Transom, 2000).

entity (in this case a ship) (Stewart, 1996). This implementation was forced to pass data regarding behaviors of the mounted human via the NPSNET high-resolution network channel, an additional network resource that was used to send non-DIS information. It was outside his research scope to mount humans so that they might be controlled as units. LT Jeffrey Halvorson USN, mounted human entities to more realistically control a submarine (Halvorson, 1997). However, this work was also only concerned with individual entities and also used the high-resolution network to pass behavior information.

F. EXTENSIBLE MARKUP LANGUAGE (XML)

The Extensible Markup Language (XML) is a new technology based on retaining the strengths of the Standard Generalized Markup Language (SGML) and the experiences learned through the development of the Hyper Text Markup Language (HTML). It designs text formats for structured data sets and new languages that are extensible, unambiguous, license-free, platform-independent, internationalizable and easily generated and read. The development of XML began in 1996, and it became a World Wide Web Consortium (W3C) recommendation in 1998. Through its use of tags and attributes, XML appears to be similar to HTML as seen in Figures 13 and 14. However, while HTML defines its tags and attributes, XML leaves their definitions to the designer and interpretation to the application that reads it. In fact, XML only uses its tags and attribute values to delimit data. While generally larger than a file in binary format, XML files are presented in text file format, which tend to compress well due to its regularity.

```
<FONT color="GREEN">DEF</FONT><B>='
</B><FONT color="TEAL">NancyTeamAddRoutes_EspduTransform</FONT><B>'
</B><FONT color="GREEN">address</FONT>
<B>='</B><FONT color="TEAL">"224.2.181.145"</FONT><B>'
</B><FONT color="GREEN">applicationID</FONT>
<B>='</B><FONT color="TEAL">1</FONT><B>'
</B><FONT color="GREEN">entityID</FONT>
<B>='</B><FONT color="TEAL">40</FONT><B>'
</B><FONT color="GREEN">marking</FONT>
<B>='</B><FONT color="TEAL">"NancyTeam  40"</FONT><B>'</B>
```

Figure 13. Example HTML Code For a "Pretty Print" Version of an X3D Fragment Produced by X3DToHTML.xsl. From (X3D, 2000).

```
<Group>
   <children>
     <EspduTransform DEF="NancyTeamAddRoutes_EspduTransform"
       address="&amp;quot;224.2.181.145&amp;quot;"
       applicationID="1"
       entityID="40"
       marking="&amp;quot;NancyTeam  40&amp;quot;"
       nodeTypeHint="Transform"
       port="62040"
       readInterval="0.25"
       siteID="0"
       traceColor="0 0 1"
       traceJava="false"
       traceOffset="0 3 0"
       traceSize="1 1 1"
       translation="-10 –200 2240">
          <children>
```

Figure 14. Example XML Code For An X3D Fragment.

Even though it is not specifically meant to be read by a human, having an XML file as a text file eases debugging and allows human modification using any text editor. Zip or tar archiving utilities can be used to compress XML files, so generally the more verbose file format of XML is not problematic. Finally, as XML matures as a technology, it carries with it a large family of supporting technologies and helpful optional modules that help define tags and attributes or task guidelines. (Bos, 2000).

G. EXTENSIBLE STYLESHEET LANGUAGE (XSL)

> Absent a stylesheet, a processor could not possibly know how to render the content of an XML document other than as an undifferentiated string of characters. XSL provides a comprehensive model and a vocabulary for writing such stylesheets using XML syntax. (Adler, 2000).

The Extensible Stylesheet Language (XSL) is used to express stylesheets and is a member of the family of XML technologies. An XSL stylesheet is used to express how

the structured content of an XML file (or class of XML files) is to be presented, or how the designer intended that it be presented. The XML source can be "styled" into or for almost any medium for presentation, such as a palm-top computer, cellular phone, a window or page in a web browser, or a printer for a hard copy of the output.

The conceptual XSL processing model, as shown in Figure 15, consists of two aspects, tree transformation and formatting. Tree transformation essentially takes the source tree (the source XML file) and builds a result (or element and attribute trees) tree from it. In the result tree, a formatting object (fo) is represented by an XML element and properties are represented by XML value-attribute pairs. The XSL stylesheet contains the rules to construct the result tree, in the form of patterns to compare against the source tree and a template to construct a portion of the result tree. Formatting is conducted by a formatter. Formatting interprets the result tree in the manner intended by the designer of the XML/XSL pair, and prepares the content to be presented in the desired manner. Formatting reads the formatting objects from the result tree and converts them into a new tree (of geographical or spatial area) called an area tree. The formatting objects represent particular formatting behaviors such as paragraph blocking with properties such as font and color. The conversion of the result tree now provides an abstract model of the presentation. The resulting abstract model is then rendered into the medium of its desired presentation whether that is a browser window, a cellular telephone display screen or paper (Adler, 2000).

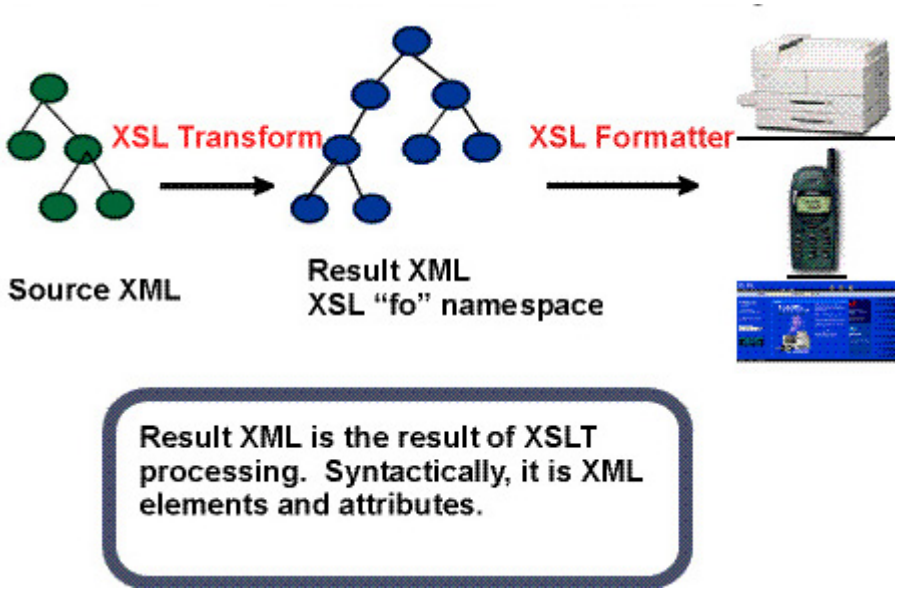


Figure 15. The Two XSL Processes: Transformation and Formatting. From (Adler, 2000).

H. EXTENSIBLE 3D (X3D) SPECIFICATION AND THE X3D-EDIT AUTHORING TOOL

> The X3D Task Group is designing and implementing the next-generation Extensible 3D (X3D) Graphics specification. We are expressing the geometry and behavior capabilities of the Virtual Reality Modeling Language (VRML 97) using the Extensible Markup Language (XML) (X3D, 2000).

The X3D Specification, in its use of the XML family of technologies, is a redesign of the fundamental syntax of VRML scenes. It defines an XML Document Type Definition (DTD) tag set which allows a user to form and validate a scene graph to describe 3D content. None-the-less, X3D maintains a strong similarity to VRML 97 Standard in its node and structure. It provides the user to generate VRML 97 on the fly and allows behaviors to react to user input. It maintains the viewpoint animation and navigation assistance provided by VRML 97. Additionally, it supports multiple

extensions such as GeoVRML, the H-Anim specification and the DIS networked behavior protocols. (Brutzman, 2000).

The X3D Specification is supported by the X3D-Edit authoring tool, which is implemented as a customized interface for IBM's XML editor Xeena (IBM, 2000). A screen shot of the X3D Edit GUI is shown in Figure 16. This authoring tool allows the user produce validated scene graphs with (essentially) error-free editing. Once directed to do so, the authoring tool calls a supporting XSL stylesheet which "formats" (i.e. translates) the XML file into VRML 97 and then "renders" it through a web browser with a VRML plug in. This goes a long way towards fulfilling the Web3D Consortium's mandate to make 3D graphics ubiquitous on the Web.

I. INTELLIGENCE PREPARATION OF THE BATTLEFIELD (IPB)

> Since war is a conflict of opposing wills, we cannot make decisions in a vacuum. We must make our decisions in light of the enemy's anticipated reactions and counteractions, recognizing that while we are trying to impose our will on the enemy, he is trying to do the same to us. (MCDP 1, 1997).

The intelligence preparation of the battlefield is a four-step continuous process of analyzing the enemy threat and environment within the confines of a specific geographical location. From the analysis and products produced from the IPB process, a battlefield commander can selectively apply and maximize his combat power at the critical time and space on the battlefield. The IPB process helps to identify facts and assumptions about the battlefield environment and threat, provides direction and

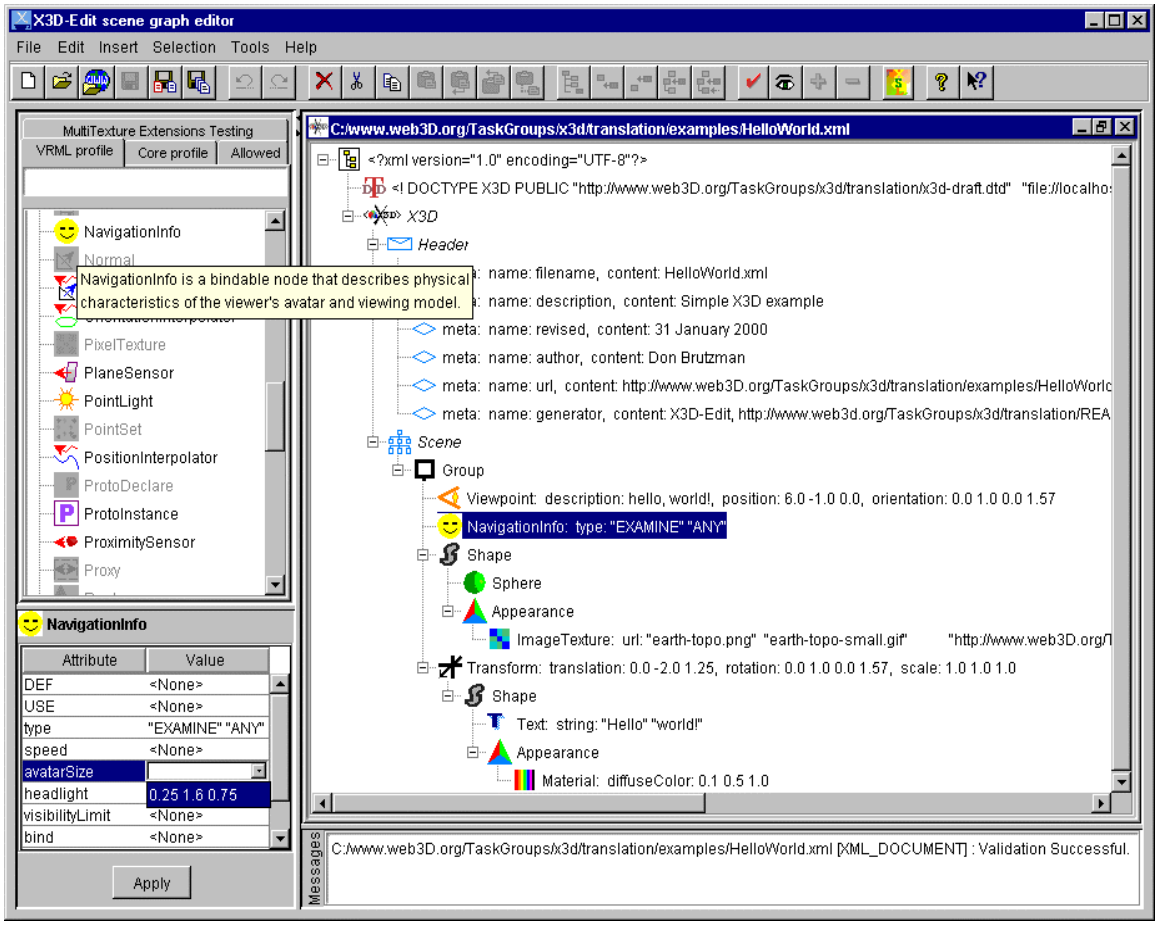
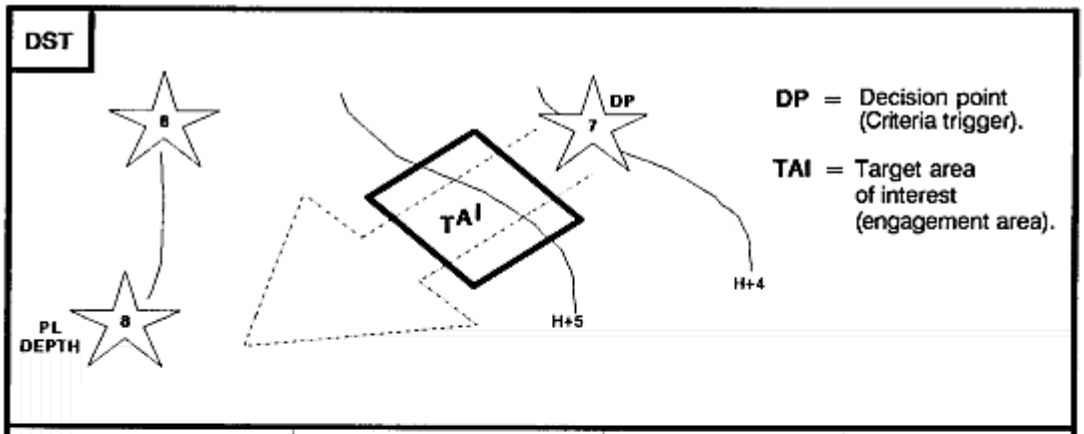


Figure 16. Screen Shot of the X3D-Edit Scene Graph Editor. From (X3D, 2000).

synchronization to the intelligence collection plan and contributes to complete staff synchronization. The process is designed to be used at all levels down to the individual soldier level. The lower the level of use, the less formalized and detailed these processes are and limited (if any) formal products are produced. As the size of the unit increases, with subsequent increase in the size of a commander's staff, the level of detail in analysis and product generation correspondingly increase. The four steps of the process are: (a) Define the battlefield environment, (b) Describe the battlefield's effects, (c) Evaluate the threat, and (d) Determine threat Courses of Actions (COAs). Products from the process

may include (but are not necessarily limited to or required, based on an element's size) modified obstacle overlays, threat doctrinal and situational templates, detailed studies on the enemy's doctrine and equipment, and comprehensive sets of possible enemy COAs, and in conjunction of the decision making process help develop battlefield operating system (BOS) synchronization matrices and decision-support templates (DST).

The BOS synchronization matrix and the DST are compiled from the results of war gaming, where friendly COAs are gamed against enemy COAs through a process of action, reaction and counteraction. During the action, reaction and counteraction process of wargaming, the commander provides decisions based on the current situation. These decisions are recorded graphically in the DST and usually in text in the supporting BOS synchronization matrix. Information recorded include the decision criteria (what was the cause of the decision?), a friendly force action or response, a decision point (DP -where geographically and/or temporally the decision must be made, represented by a star), target areas of interest (TAI - where the effects of friendly force actions are synchronized), named areas of interest (NAI - locations from which information is needed to confirm or deny the decision criteria) and possible changes or nominations to the high priority and high value targeting lists. See Figure 17 for a combined BOS synchronization matrix and DST and Figure 18 for a DST imposed on a map with operational graphics. (FM 34-130, 1994).



DP		No. 6	No. 7	
Time		H+2	H+4	
Decision Criteria Trigger (Enemy Action)		Enemy defenses extend East past PL DEPTH (in Bn strength or more).	MRR CATK (Tk Co)	
(BOS) FRIENDLY ACTIONS	Intelligence	UAV to NAIs 21, 22, 23, 24, 25, and 26.	2-9 CAV Collects	
	Maneuver	Shift from Bdes, abreast to Bdes in column on Axis EILEEN.	Attack Helos to TAI 7 AT H+5	
	Fire Support		H+5 DPICM to TAI 7 Jam MRR C^2	

Figure 17. Combined BOS Synchronization Matrix with DST. From (FM 34-130, 1994).

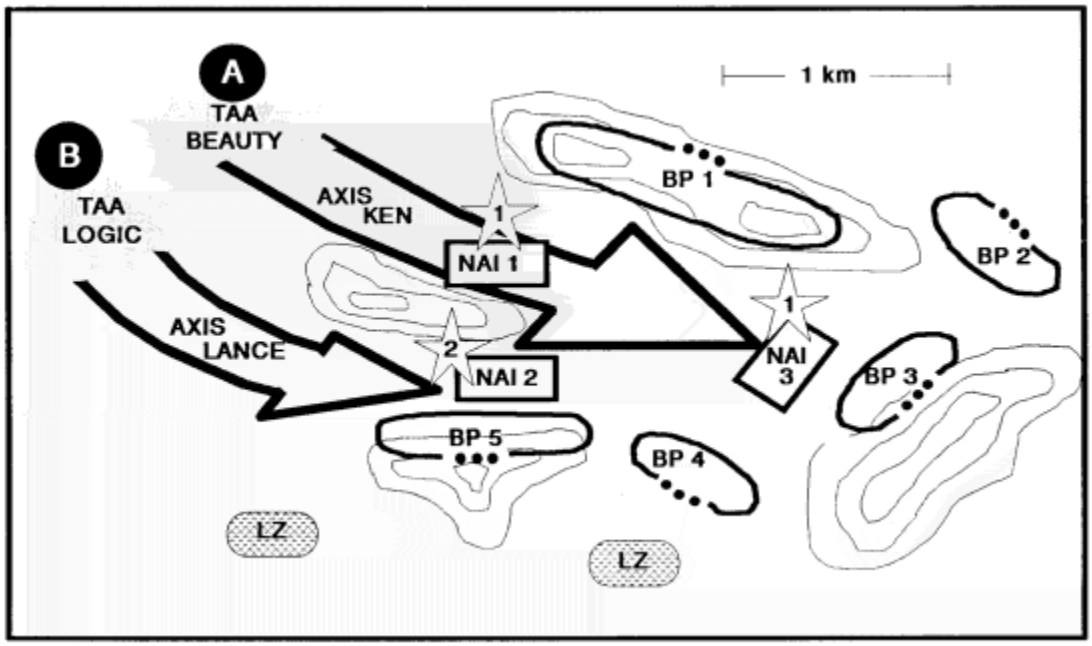


Figure 18. DST Superimposed Over a Map with Operational Graphics. From (FM 34-130, 1994).

J. SUMMARY

This chapter details the DIS-Java-VRML framework, the H-Anim 1.1 Spec and the humanoid models used in this research. It examines the new and developing technologies, like XML, XSL, X3D and the X3D-Edit authoring tool, used in support of this thesis project. Additionally, it reviews background work in large-scale virtual environment research and briefly discusses the IPB and decision process.

III. PROBLEM STATEMENT

A. INTRODUCTION

This chapter defines the problem statement of this research, offers a proposed solution, specifically addresses the focus of the research, and then examines design issues considered in the development of the model implementation.

B. PROBLEM STATEMENT

With the fall of the former Soviet Union in 1990, the assigned missions of the United States (US) military have undergone a profound paradigm shift. The advent of multinational narco-terrorism and organized crime, the resurgence of nationalism and the continuance of radical religious fundamentalism and regional power aggression has led to a higher level of US military deployments involving ground forces putting "dirty boots on the ground." There is the realization that there is a shortfall of decision and visualization aids to support these increased deployments, particularly in SOF missions (Rand, 1994) and Military Operations in Urban Terrain (MOUT) (Pew, 1998).

Unfortunately, support in the modeling and simulation community has not yet answered the emerging needs of this new world reality. Thus, there exists a need for the development of high-resolution 3D visualization tools for tactical mission planning. Technical problems include model interoperability, georeferenced coordinate systems, retaining double-precision accuracy while using single-precision variables, network

latency, network quality of service and network reliability. Any satisfactory solution must address all of these problems simultaneously.

C. PROPOSED SOLUTION

The proposed solution for these multiple challenges is to develop a proof of concept model for a networked 3D visualization tool within an existing LSVE. The tool must contain articulated human entities, aggregated into small units, capable of showing physically based human motion as well as the performance of realistic group behaviors, under the control of a single or multiply users, as required. The system must be extensible, platform-independent and worldwide deployable to support current unfulfilled and emerging needs.

D. RESEARCH FOCUS

The focus of the research is to produce aggregated groups of virtual humans (using the Nancy.wrl file), which display realistic group behaviors, as defined by dismounted infantry tactical formations and activities, in the networked 3D virtual environment provided by DIS-Java-VRML CTF. Human entities must further be able to disaggregate from a group and return to independent control. Additionally, the need to rapidly develop human content with interchangeable geometries and behaviors is examined as a requirement to effectively support such a visualization tool.

E. DESIGN CONSIDERATIONS

The most significant design consideration was how best to implement the aggregation and disaggregation of the networked human entities. Multiple

implementations were designed, tested, examined and reviewed before an implementation was chosen for the model. The selected implementation uses relative coordinates from a non-physical aggregation entity passed over the network via the DIS protocol. The file is composed as a VRML scene graph with the DIS-compliant aggregation entity and, wrapped within separate transform nodes, the human entities as scene-graph peers. Aggregation and disaggregation is handled through sophisticated, run-time VRML scene graph switching which executes the addition and deletion of VRML routes.

Other major design considerations were: implementing the Nancy.wrl file, with its accompanying motion library, in a DIS-compliant format and then including it within CTF; defining the group behaviors for the aggregated entities to execute; designing the interface (both graphical and logical) for the single aggregated human entities; and developing a new behavior (e.g., kneeling) to include in the motion library to demonstrate the desired interchangeability of behaviors and geometries.

F. SUMMARY

This chapter defines the problem addressed by this thesis research. It discusses the proposed solution and the specific focus of this research. It addresses the method chosen to implement the problem of entity aggregation and disaggregation.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. INITIAL DEVELOPMENT EFFORTS

A. INTRODUCTION

This chapter examines, in detail, the challenges involved making the scene Nancy.wrl DIS-compliant, designing a single-user, single-entity interface to control the DIS-compliant Nancy, and then including Nancy in the CTF framework as well as implementing the file within the Herrmann Hall terrain database. It discusses an extension to the motion library through the addition of animation interpolators to create a kneel behavior. Additionally, improvements made within the CTF world regarding identification and removal of extra sound threads plus a brief discussion of VRML plugins for web browsers are presented.

B. NANCYS IN CTF

The first challenge of this research was to modify and incorporate the canonical H-Anim 1.1 specification exemplar file, Nancy.wrl, into an acceptable DIS-Java-VRML entity, and subsequently incorporate it into the CTF game. First, the .wrl file had to be modified to accept DIS packets along with the accompanying Java scripting and logic. Second, an interface to control the DIS-compliant Nancy, consisting of a graphical user interface (GUI) control panel and an action interpreter, both written in Java, was designed and implemented. Subsequently, the animations of the motion library were incorporated in the interface so as to logically occur in CTF as human behaviors. Finally, when this

was accomplished, experiments in moving the DIS-compliant Nancy to a different LSVE and controlling multiple Nancys were conducted.

1. DIS, Animations and Scripts

The initial design concern in implementing Nancy in the DIS-Java-VRML framework was the multiple nesting of prototypes. The original Nancy already has one layer of PROTO nesting by using Joint and Segment within Humanoid. In order to make it DIS-compliant, the PROTOs defining the geometry needed to be further nested within the EXTERNPROTO of the EspduTransform node. However, in execution, the browser was able to read all PROTO declarations, and was able to render the file. A rotation of 0.5 * Π (90 degrees counterclockwise) around the positive Y axis (up direction in VRML) had to be placed around the instance of the Humanoid PROTO in order to appropriately register Nancy's movement within CTF and other DIS-Java-VRML environments.

Nancy's movement library execution model also had to converted from the original model, where behaviors were performed through the use of a VRML touch sensor (mouse click) on a heads up display, to a suitable model for execution in CTF. Since the goal of this thesis was to implement realistic behaviors, this involved matching behaviors to environmental input; such as movement behaviors executed based on the magnitude of the entities' linear velocities. Other behaviors are implemented through user command via a control panel. To execute this model, a behavior articulated parameter was implemented in the DIS-compliant Nancy.wrl file that was subsequently fed from the single-entity interface developed for Nancy. The articulated parameter in the

interface enumerated the behavior library and sent this via DIS packet to the EspduTransform Java (language) Script file, which communicates with the VRML file. The input from the Java (language) Script file is then communicated within the VRML file via routes to an internal VRML Script node containing a ECMAScript. The ECMAScript translates the enumerated behavior and passes the translated behavior to interpolators within the VRML file, which then correctly renders the file within the scene. See Figure 19.

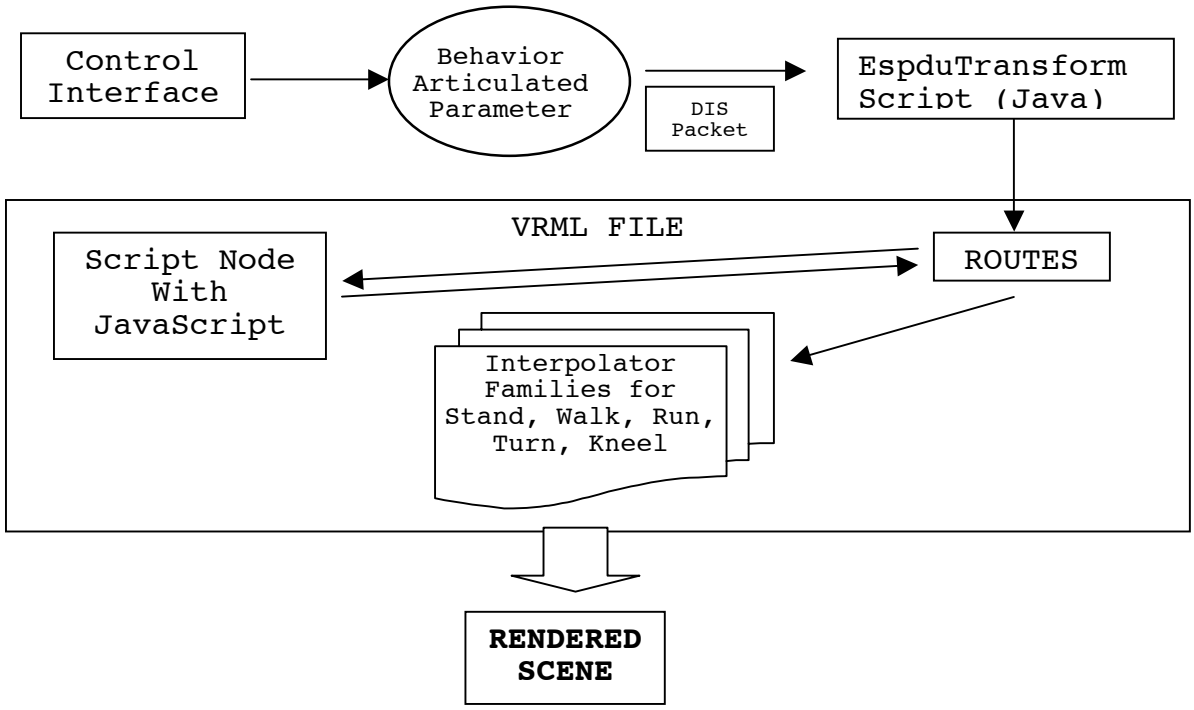


Figure 19. Flow of Human Behavior Implementation via an Articulated Parameter Executed Within the DIS-Java-VRML Framework.

2. Single Nancy Interface

The single Nancy interface consists of the GUI control panel, as seen in Figure 20, and the logic, in the form of an action interpreter, which translates user input from the panel and sends it to the VRML scene in a form in which it can be understood and rendered. The GUI is implemented as a Java swing component jFrame, which runs independently of the VRML scene. This allows it to be moved, resized or hidden to maximize the user's experience. The panel itself offers simple navigation controls and information to allow control of speed and heading. It also allows the user, through button control, to fire a weapon, kneel, come to a full stop, reset to its start position or restart as a new player. The animation behaviors are initiated in the action interpreter either as indicated by achieving a threshold magnitude of linear velocity, or in the case of kneel, by user input command. The action interpreter runs iteratively as a child thread of the control panel, forms DIS-compliant ESPDU packets and transmits them to the network to be handled by the EspduTransform Java (language) Script class approximately every half a second. The VRML scene reads updates more frequently than the send interval to ensure low latency, and these intermediate posture updates are handled by the dead-reckoning algorithms provided in the EspduTransform Java (language) Script class.

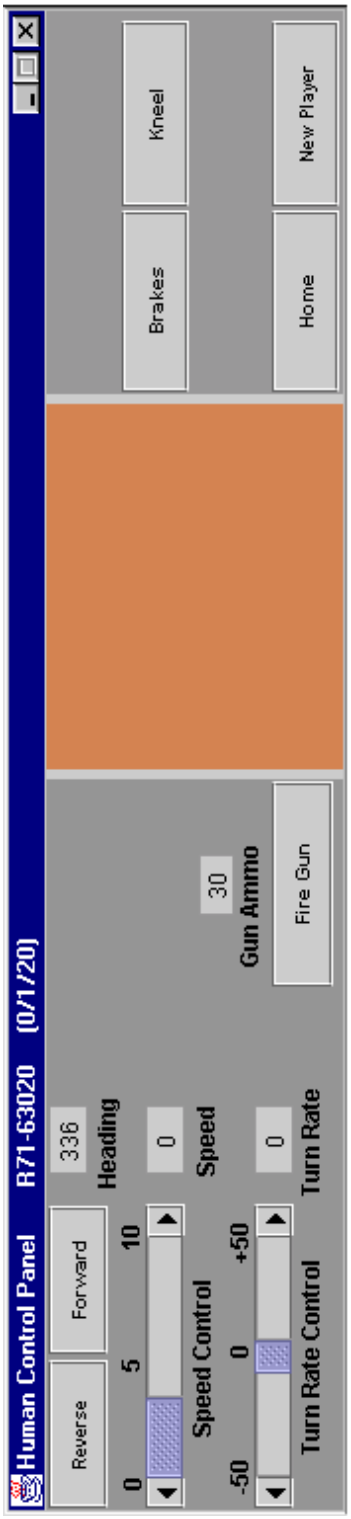


Figure 20. Single Human Entity, Single User Control Panel. The orange center panel is reserved for future work on situational awareness maps.

3. Nancy in Herrmann Hall

To demonstrate the interchangeability of geometries and behaviors between worlds, the DIS-compliant Nancy file was inserted into a world containing Herrmann Hall as shown in Figure 21. The Herrmann Hall model (NPSNET, 2000) was created by John Locke for NPSNET IV using the authoring tool Multigen Creator (Multigen, 2000) and subsequently exported as a VRML file. It contains approximately 10,000 polygons and 92 different textures in its original format. After the Nancy file was appropriately registered, she was able to move around this world, external to the building, exhibiting her motion library, in the DIS XY plane as seen in Figure 22. Movement inside the building fell outside the scope of the demonstration due to complex collision detection.

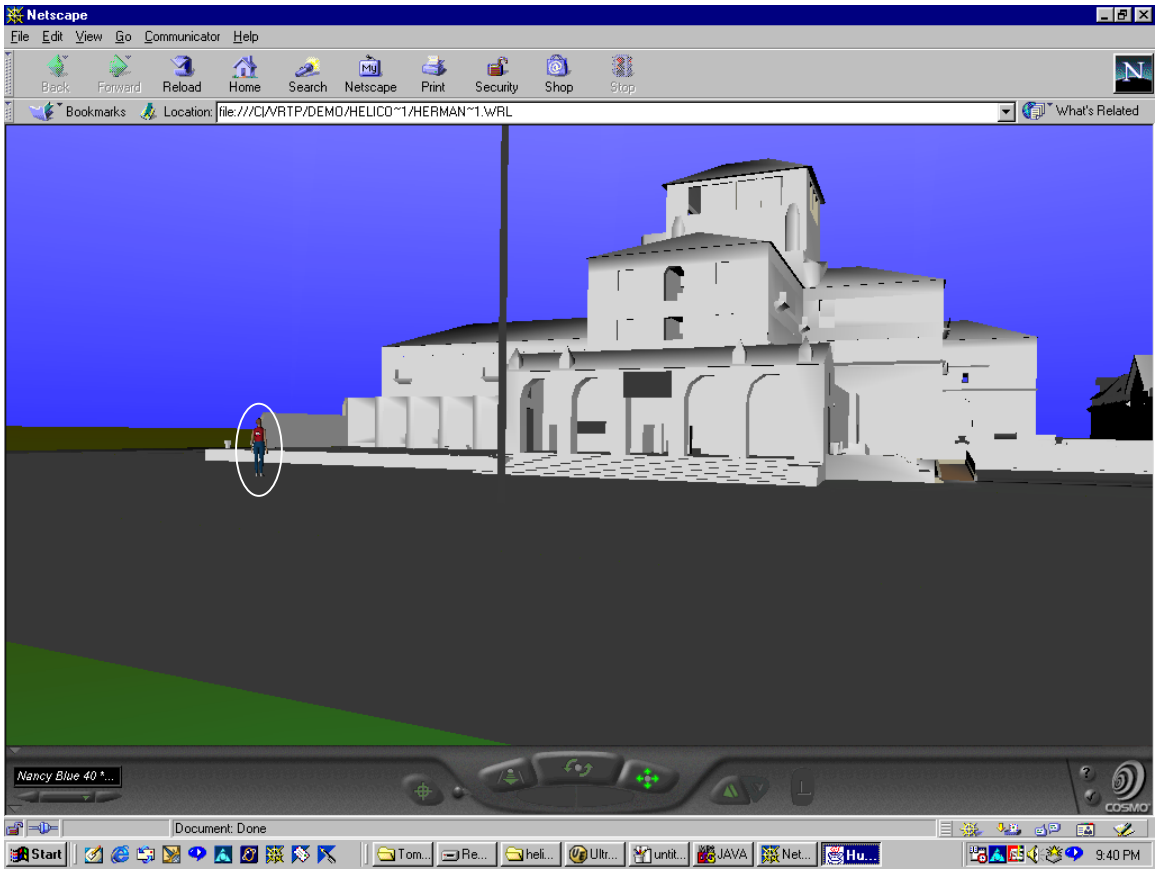


Figure 21. Nancy in the left foreground outside of Herrmann Hall (circled in white), seen from 25 meters. Note the flagpole in the center.

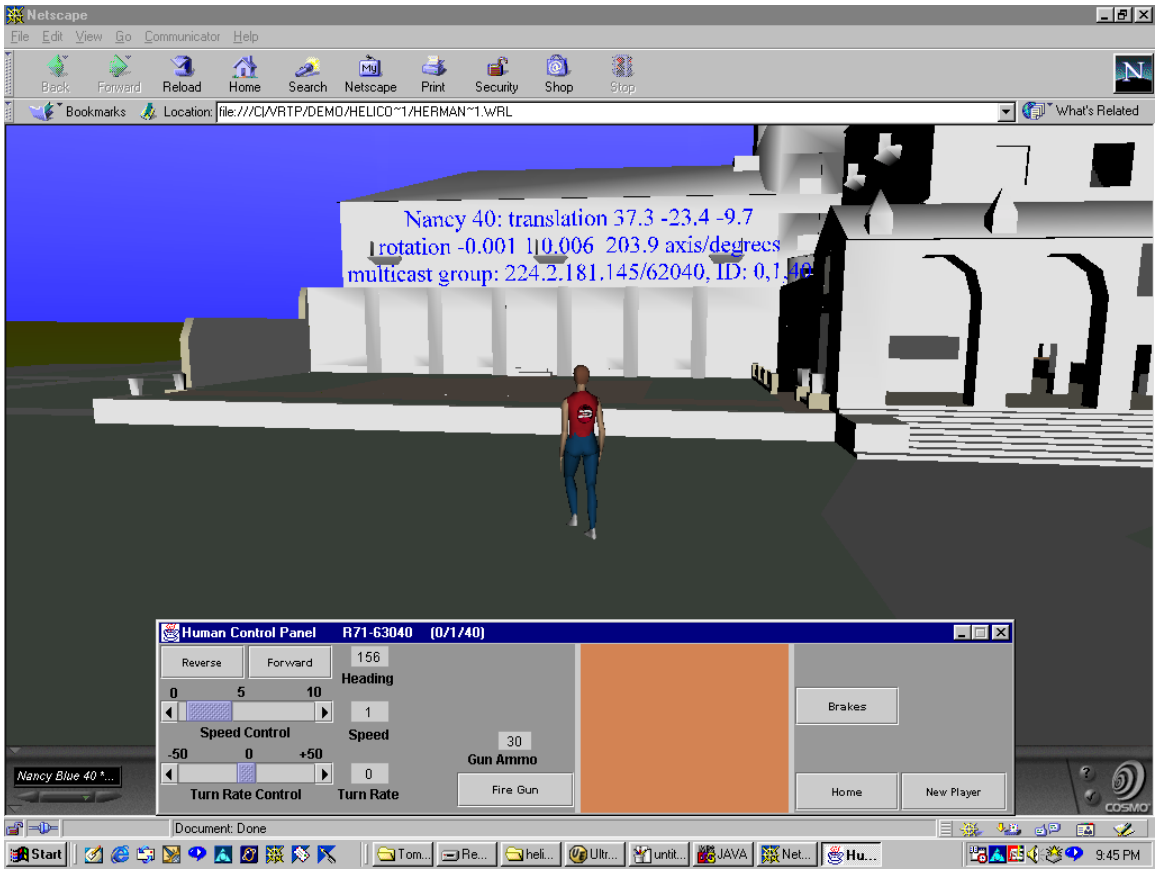


Figure 22. Close view of Nancy moving in front of Herrmann Hall, with trace values and control panel shown.

4. Adding Behaviors to the Motion Library

To illustrate the ability to extend the motion library, the "kneel" behavior was developed. The new behavior was developed using the Parallel Graphics Internet Character Animator (ICA) VRML development tool, shown in Figure 23. The ICA provides powerful visual support for professional quality animations and is H-Anim compliant. The new behavior was first included in the original Nancy.wrl file, shown in Figure 24, and activated by a Touch Sensor. Then the behavior was modified to support

the DIS-compliant Nancy by incorporating a move to kneel from stand behavior as well as a kneel behavior, shown in Figure 25.

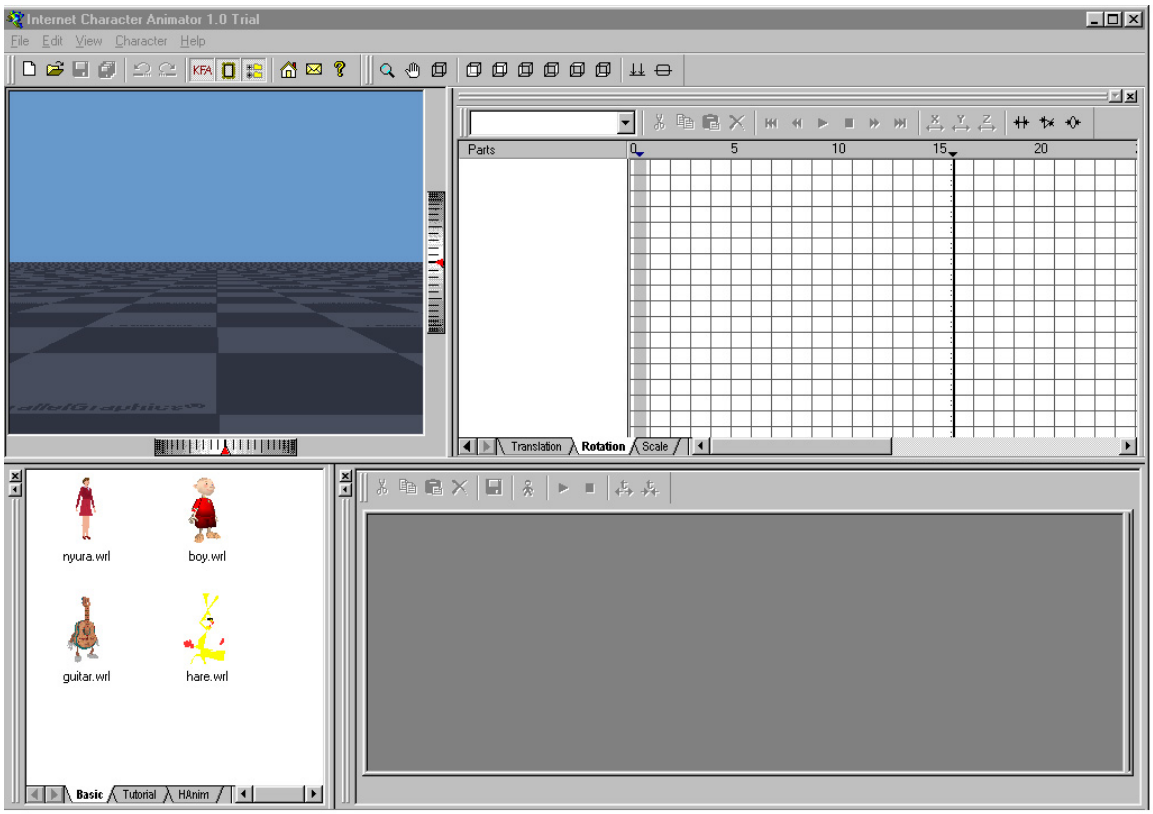


Figure 23. Parallel Graphics Internet Character Animator. From (Parallel, 2000).

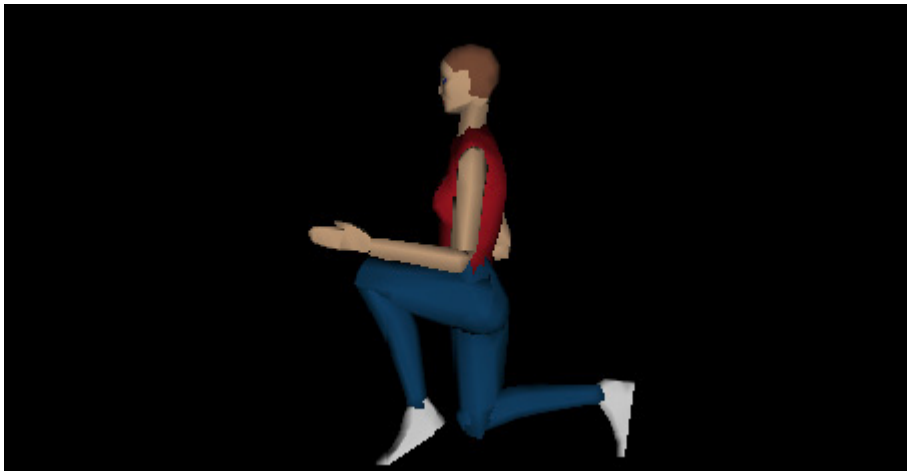


Figure 24. Nancy.wrl Demonstrating Kneel Behavior.

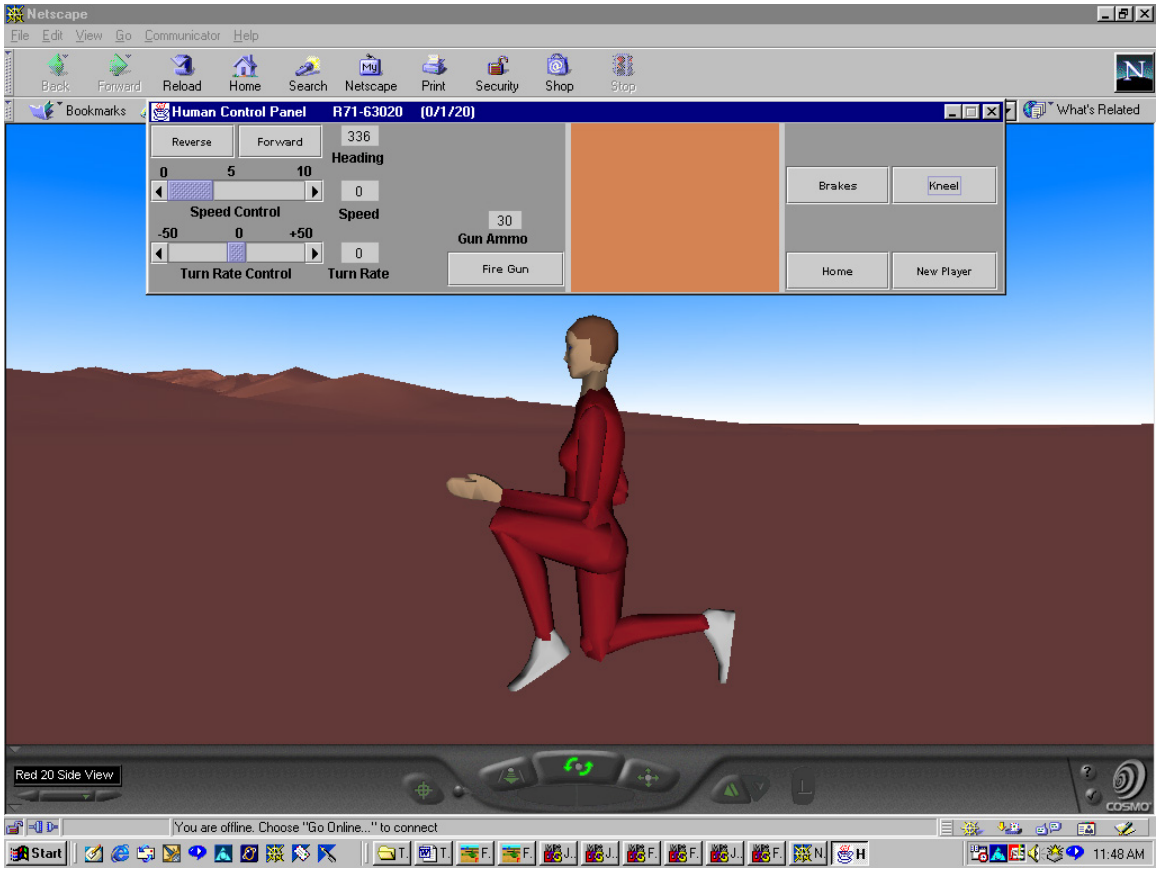


Figure 25. DIS-Compliant Nancy demonstrating Kneel behavior through use of the control panel Kneel button.

5. From One to Many

Once the Nancy file had been made DIS-compliant and shown to be interchangeable between worlds, the more complex problem of how to scale up to several Nancys while maintaining a single controller for these multiple entities had to be addressed. This problem had network, scene graph and mathematical transform aspects working in concert. Several non-interactive two-entity and three-entity implementations were developed, but in all these early attempts, network latency led to an unacceptable level of hysteresis or "popping" from the non-lead entities, as they constantly tried to reach their correct position in the virtual world. This problem is further examined and resolved in Chapter VI.

C. THREAD PROBLEMS AND SOUNDS

Even though the subject of sound nodes and the accompanying threading problems associated with them is not directly part of this research, the isolation, identification and solution of this problem, existent from its creation in 1996, increased the usability of CTF to an extent to allow my research (and that of other students) to continue on this platform. DIS-Java-VRML and CTF are, by their very nature, required to be multithreaded. Unfortunately, the thread count, and subsequent memory lock, occurring from running CTF was large enough not only to significantly and adversely effect the display frame rate, but essentially made the program unexecutable for greater than five minutes due to the drain on virtual memory.

At first, the thread problem was not readily apparent, and the root cause even further obfuscated. Upon systematic examination of computer system monitoring utilities across various operating systems, we discovered that an unexpectedly large number of threads were consistently being generated by the browser while running CTF. Subsequently, we began to examine different components within the source code base that might be adversely effecting the threading.

The next break came while testing CTF on a Windows98-based machine without a sound card. Without the sounds, significant increases in performance and significant reductions in thread count were experienced. Knowing where to look, we began to systematically scrub the code for sound nodes, and then design, code and test different implementations to reduce the number of sound nodes while maintaining the same level of fidelity spatialized sound within CTF. We were able to develop an implementation through careful combination of source files, using file inlining and defining (DEF) and using (USE) Sound node sources, such that we were able to reduce the thread count by over 180 threads, providing a 50% reduction in overall thread count. This made CTF a stable and reliable research platform once again.

Concurrent with these activities, we queried members of the Web3D Consortium for possible sources of the problem as well as identified this problem for future conformance testing for the new X3D specification. There were no specific reference to a problem such as we encountered in CTF in previously published materials, specifications or Frequently Asked Questions (FAQ). However, through electronic mail with David

Chamberlin, formerly of Silicon Graphics, Incorporated (SGI), where CosmoPlayer was developed, we did identify the suspected overall root cause.

The CosmoPlayer

> …audio/video library was designed to be a very extensible, pluggable library with the ability to do very tight synchronization and the ability to be used in a highly multithreaded environment (Chamberlin, 2000).

This causes CosmoPlayer to provide robust performance in exchange for a very "heavy weight" underlying structure. Availability of open source for future browsers will greatly improve our ability to diagnose and correct such difficult problems.

D. BROWSER COMPARISON

There are three major VRML browsers, CosmoPlayer 2.1.1, Cortona VRML Client and Blaxxun Contact 4.4, currently available. All three work inside the Netscape and Internet Explorer family of browsers. Each are available to download free of charge and all offer advantages and disadvantages.

1. CosmoPlayer 2.1.1

CosmoPlayer was originally developed by SGI, with the latest version 2.1.1 released in December 1998. CosmoPlayer has been the browser standard in relation to conformance testing. However, it has numerous identified program bugs and unfortunately, 2.1.1 was the last work done on the source code because SGI sold it to Platinum Technologies (and subsequently resold to Computer Associates Inc.). Since the CosmoPlayer source code is stagnant, Blaxxun and Cortona, who have continued to develop their products, offer a higher level of support for updated technologies and

advanced features. CosmoPlayer is the only browser that currently supports the Java network security model required to run DIS-Java-VRML. Unfortunately, it also contains the sound thread problem and also does not conform to the VRML97 specification due to its inability to render H-Anim 1.1 specification humanoids using EXTERNPROTO definitions.

The control panel, shown in Figure 26, provides interactivity through viewpoint (bottom left) and navigation controls (bottom center). Preferences can be set and a VRML console can be called from the check button (bottom right). Gravity can also be applied and each rendering comes with a default headlight or light from the viewpoint so that lighting does not have to explicitly be built into the file.

2. Cortona VRML Client

Parallel Graphics, the maker of the Cortona VRML Client, shown in Figure 27, have been the most active company in developing Web3D tools recently, and seemingly is the most committed to making a profit from these technologies. The Cortona VRML Client was released for the Apple Macintosh on 5 September 2000. They are also developing a VRML Client for Windows CE for handheld devices. They have developed a series of four VRML authoring tools (VRML Pad, Internet Space Builder, Internet Scene Assembler, Internet Character Animator) and are preparing to release a fifth tool to optimize 3D scenes. Cortona is currently the smallest VRML browser and claims to be the fastest.

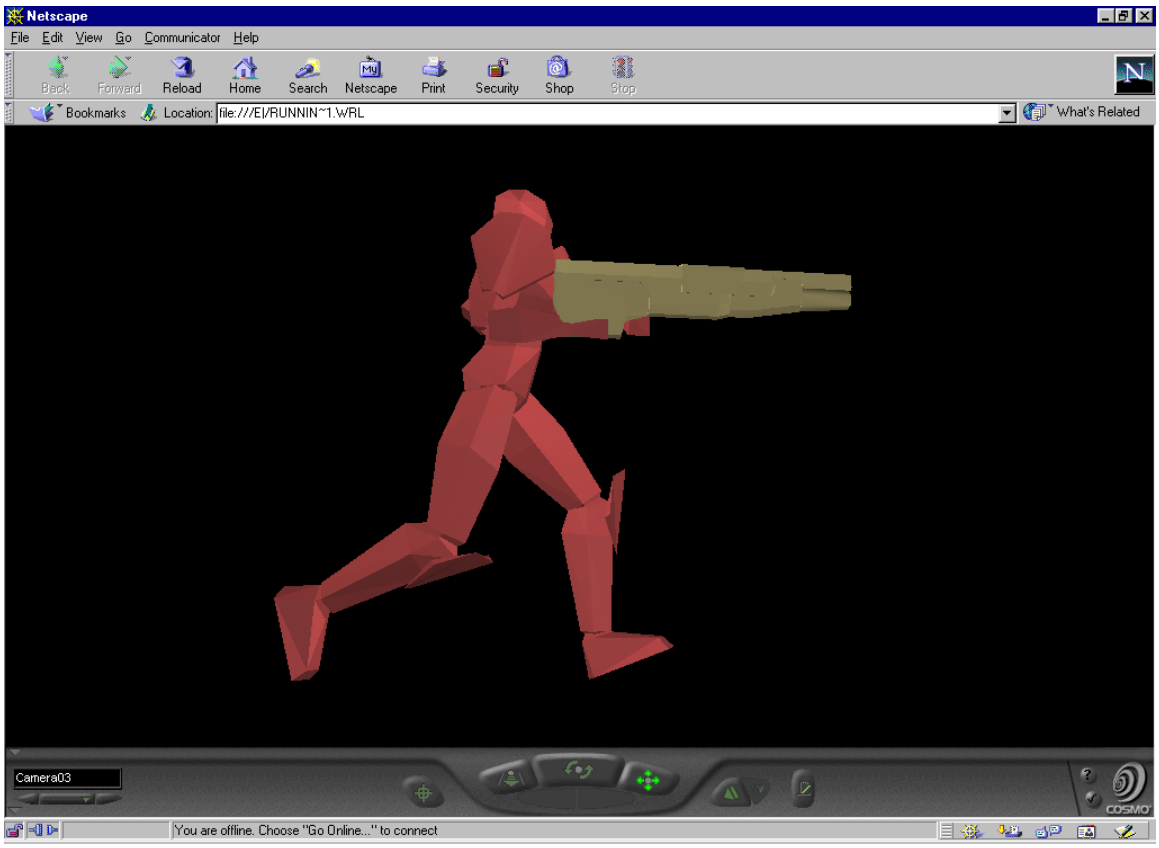


Figure 26. CosmoPlayer 2.1.1 showing the model RunningWomen.wrl. From (ZdNet, 2000).

Cortona provides standard navigation and viewpoint interactivity. However, Parallel Graphics has added extensions to define and support Spline geometries and NURBS, support REAL Audio and Video and Macromedia Flash for inclusion as movie textures, as well as extended event generation to support drag and drop and keyboard input. On the cutting edge of technology, the latest VRML client is optimized for the Intel Pentium III processor and provides full support for Software renderings and hardware accelerators DirectX and OpenGL. Unfortunately, at this time, Cortona does not support the DIS-Java-VRML security model, but it does support the H-Anim 1.1

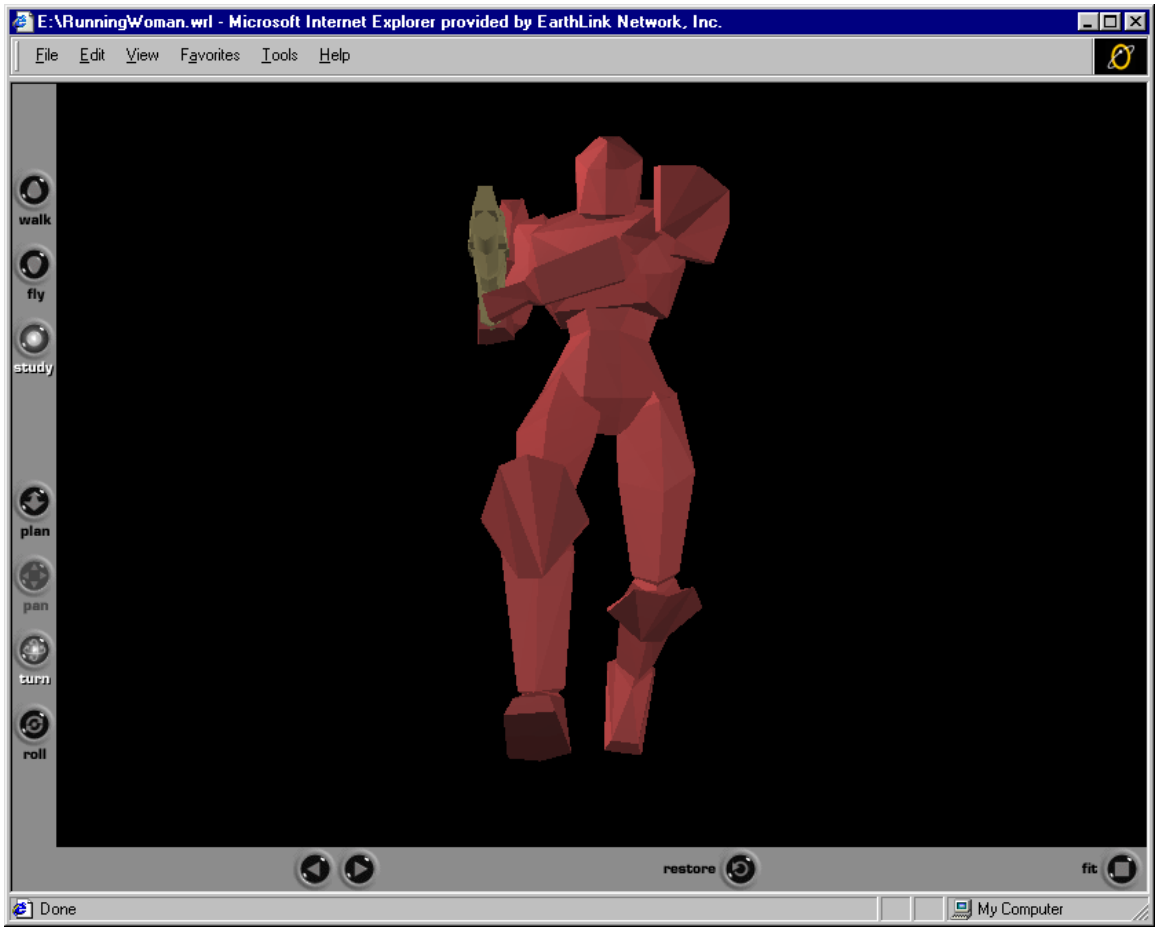


Figure 27. The Cortona VRML Client showing the Model RunningWoman.wrl. From (ZdNet, 2000).

specification native tag set translation from developed for X3D-Edit. The DIS-Java-VRML working group continues to encourage Parallel Graphics to enable the next release of the Cortona VRML Client to support the DIS-Java-VRML security model.

3. Blaxxun Contact 4.4

Blaxxun, began in 1995, is an internationally based company, bringing interactive 3D for virtual worlds across the web. Blaxxun Contact 4.4, shown in Figure 28, is their latest update in their Contact VRML browser family and was released 2 August 2000.

Blaxxun has experimented with integrating new technologies into their VRML browsers, such as including a NURBs specification and most recently a sample Cells and Portals implementation for 3D optimization. There have also shown consistent interest in rapid avatar creation with the Blaxxun Avatar Den and most recently with the Blaxxun Avatar Studio.

The control panel also allows interactivity through navigation and viewpoint control. Blaxxun also includes more technologically advanced controls through a pop up menu activated by right clicking in the browser. These include providing an avatar to represent the position of the viewer, plus wire-frame and vertices-only drawing of scenes. See Figure 29.

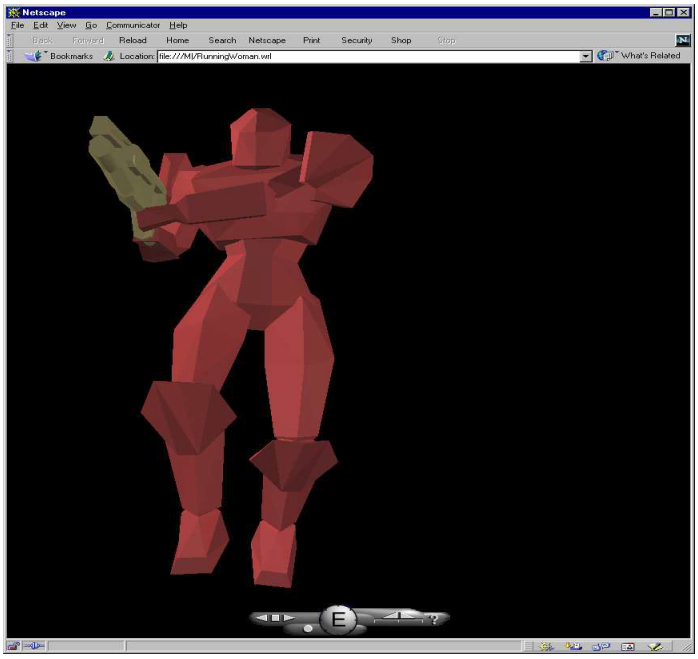


Figure 28. Blaxxun Contact 4.4 showing the model RunningWoman.wrl. From (ZdNet, 2000).

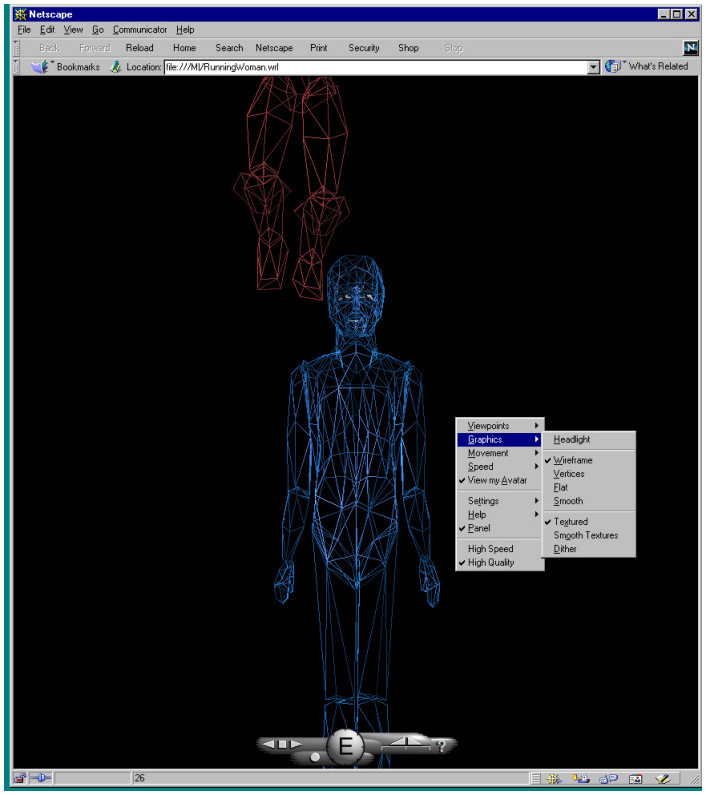


Figure 29. Blaxxun Contact 4.4 Showing Wire frame Rendering of the Blaxxun Avatar.

E. SUMMARY

This chapter discussed the challenges involved in implementing the model Nancy.wrl in a DIS-compliant format, and designing a single-user, single-entity interface to control the DIS-compliant Nancy. The inclusion of the DIS-compliant Nancy into the CTF framework and an alternate VE world, as well as extending the motion library are addressed. Additionally, improvements made within the CTF world regarding sound threads, and a brief discussion of VRML browsers are presented.

V. H-ANIM IN X3D

A. INTRODUCTION

This chapter provides a brief overview of the significance of switching to X3D when dealing with human avatars. Following this is a description of the original conversion of the existing Nancy.wrl file to XML and then its translation back to VRML. Finally, the creation and implementation of a H-Anim 1.1 Spec DTD is discussed.

B. WHY X3D?

The X3D specification is an evolutionary step forward toward the realization of the Web3D Consortium's mandate to make 3D graphical content ubiquitous on the web. The X3D-Edit authoring tool is the proof-of-concept demonstration application to leverage new and emerging technologies to further drive this mandate and make rapid, error-free 3D content generation as easy as constructing a 2D web page is today. While fully implementing and supporting the VRML97 standard, this is merely an interim step in X3D's final form. The ease in which extensions to the standard can be incorporated into the authoring tool and the control of the final output format through the XML family of technologies make it the platform from which to implement revolutionary leaps in web 3D graphics emerging in the near future. The inclusion of the extensions of the H-Anim specification, discussed in this chapter, GeoVRML specification and DIS-Java-VRML framework are blueprint exemplars from which to continue to develop and include powerful new concepts in web 3D graphics.

C. TRANSLATING NANCY

Since both the original Nancy and the DIS-compliant Nancy were written solely in VRML, the file needed to be converted into XML and then translated back into VRML to ensure compatibility forward and backward, as well as provide a stable starting point to more rapidly create human graphics content.

1. *.wrl to *.xml

Unlike the robust translation support provided by X3DToVRML.xsl, no good VRML to X3D translator yet exists. The VRML-Java3D working project and open source X3D/VRML-Java3D (Xj3D) codebase, primarily developed by Sun Microsystems, have made steps in that direction. Unfortunately, in its current implementation of its Document Object Model (DOM) X3Delement class, attributes are typed by context and not by content. While as a generalized model, this provides a powerful translation tool, it is unable to translate PROTOs as PROTOs and instead fully writes each instance of a PROTO out in an operative format for execution. Nevertheless, this partial translation provided a baseline to start from, while the rest of the translation required work by hand using the X3D Edit tool as seen in Figure 30.

2. *.xml to *.wrl

The XML to VRML conversion is handled through an XSL style sheet. In the case of the original Nancy.xml translation, the unchanged X3DtoVRML97.xsl file was used. This style sheet reads the attributes and tags from the XML file, translates

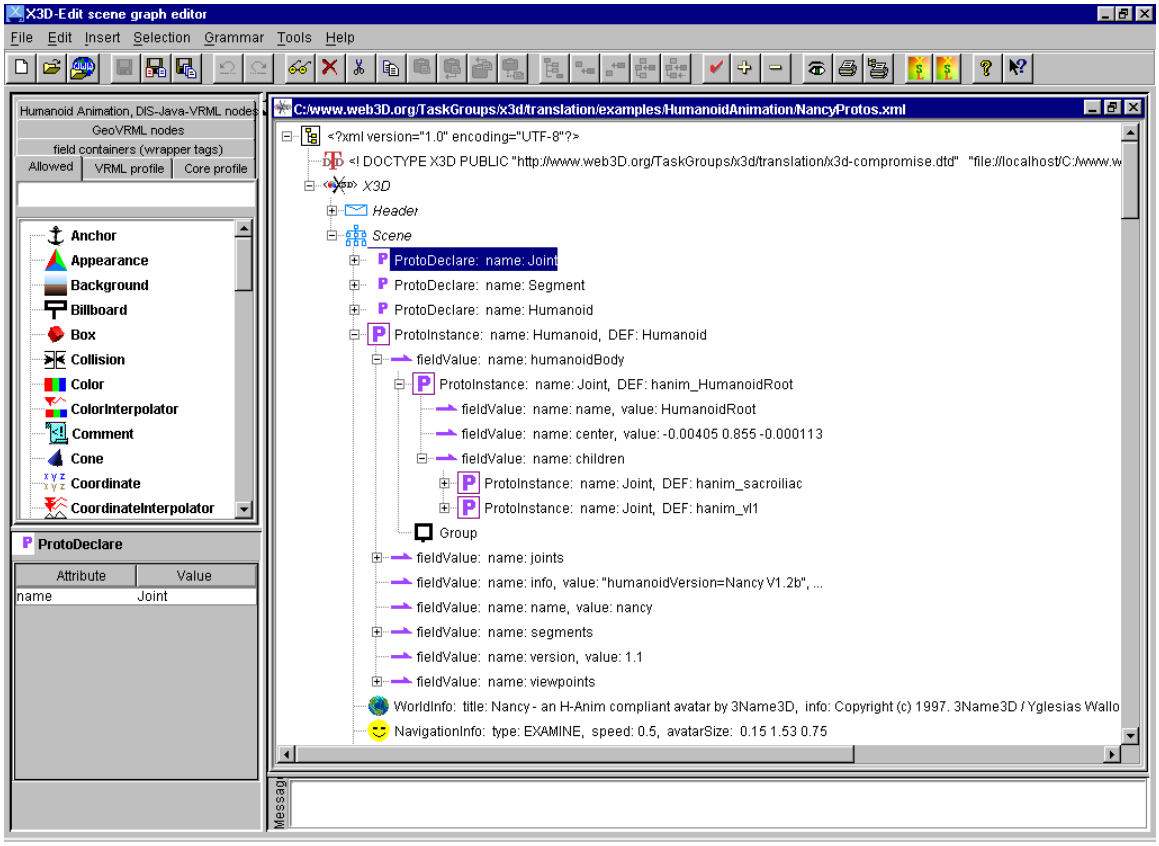


Figure 30. X3D-Edit Screen Shot of Nancy.xml.

them into a VRML97 format and launches a web browser with a VRML plug in as the renderer. It was able to read the PROTO declarations and PROTO instantiations written into the Nancy.xml file, and therefore render the VRML file appropriately in a web browser. See Figure 31 for a fragment from the X3DtoVRML97.xsl file, which determines the node type of PROTOs and EXTERNPROTOs, thus used to translate Nancy.xml back into VRML.

```
    <xsl:apply-templates />
  </xsl:template>
  <!--  ****** recurse through each of the tree node elements ******
    -->
  - <xsl:template match="*">
    <!--  figure out node type for ProtoDeclared ProtoInstance,
      ExternProtoDeclared ProtoInstance and internal DTD declaration
      tag definitions  -->
      <xsl:variable name="nodeName" select="./@name" />
      <xsl:variable name="nodeName2" select="local-name()" />
      <xsl:variable name="parentNodeName" select="../@name" />
      <xsl:variable name="parentNodeName2" select="local-name(..)" />
    <!--  ProtoDeclared ProtoInstance  -->
      <xsl:variable name="nodeType" select="local-
        name(//ProtoDeclare[@name=$nodeName or
            @name=$nodeName2]/*[local-name()!='field'])" />
      <xsl:variable name="parentType" select="local-
        name(//ProtoDeclare
            [@name=$parentNodeName]/*[local-name()!='field'])" />
    <!--  ExternProtoDeclared ProtoInstance  -->
      <xsl:variable name="EPnodeType" select="//ExternProtoDeclare
           [@name=$nodeName or @name=$nodeName2]/@nodeTypeHint" />
      <xsl:variable name="EPparentType" select="//ExternProtoDeclare
            [@name=$parentNodeName]/@nodeTypeHint" />
```

Figure 31. Fragment from X3DtoVRML97.xsl Which Determines Node Types of PROTOS. From (X3D, 2000).

3. Lessons Learned

The major lesson learned is that converting a large graphics file by hand is tedious, time-consuming work. It is far preferable to have a conversion program do the work. Suggestions and recommendations developed during the conversion experience were passed on to Rick Goldberg, Sun MicroSystems, the primary Xj3d author, for use in further development of backwards-compatibility translations. For more information on Xj3D, visit the How to Install and Compile Xj3D page at http://www.web3D.org/TaskGroups/x3d/Xj3D/HowToInstall.html. On the positive side, the conversion work validated the forward compatibility from XML to the H-Anim 1.1 spec in VRML, which

allowed the development of the H-Anim EXTERNPROTOs and the development and implementation of a native tag set for H-Anim in the X3D-Edit tool.

D. GOING NATIVE

In order to rapidly generate virtual human content in X3D-Edit, a H-Anim DTD and native tag set had to be developed and implemented. This required defining EXTERNPROTOs for the H-Anim 1.1 spec PROTO nodes in a file, defining the DTD in an XML file and then adding the appropriate code to the X3DtoVRML97.xsl stylesheet in order to allow the correct translation and format. In each case, development and support of prototype declaration versions enabled getting the code to work, and then simplified the generation of native tag support.

1. EXTERNPROTOS for H-Anim 1.1 Spec and the H-Anim DTD

The H-Anim EXTERNPROTO declarations used for the XSL style sheet translation reside in the VRML translation of NancyProtos.xml called NancyProtosTranslated.wrl. As per the VRML specification, libraries of PROTOs can be created by including multiple PROTO definitions within the same file. Specific PROTO definitions are accessed by adding a pound sign (#) and node-type name to the end of the Uniform Resource Locater (URL) used in the EXTERNPROTO declaration (Ames, 1997). Of note, the Nancy files only include the three major PROTOs of the H-Anim 1.1 spec (Humanoid, Joint and Segment Nodes), not all five (the Site and Displacer Nodes), and these three are the only EXTERNPROTOs currently defined in X3D-Edit. The

EXTERNPROTO definitions supporting H-Anim have been included in the current builds of X3D-Edit since version 1.4.

The DTD was developed from the node definitions provided in the H-Anim 1.1 Specification and implements all five of the specified H-Anim nodes. It is the list of tags that define the attributes and elements of H-Anim, and describes their relationships and their formatting. It also provides the ability to validate XML files containing H-Anim content by ensuring that the content adheres to the specified formatting and structure. See Figures 32 and 33.

```
<!ELEMENT Humanoid (
 (humanoidBody, (
      (joints,((segments,((sites,viewpoints?)|(viewpoints,sites?))?)
     |(sites,((segments,viewpoints?)|(viewpoints,segments?))?)
     |(viewpoints,((segments,sites?)|(sites,segments?))?) )? )
     |(segments,((joints,((sites,viewpoints?)|(viewpoints,sites?))?)
     |(sites,((joints,viewpoints?)|(viewpoints,joints?))?)
     |(viewpoints,((joints,sites?)|(sites,joints?))?) )? )
     |(sites,
        ((joints,((segments,viewpoints?)|(viewpoints,segments?))?)
     |(segments,((joints,viewpoints?)|(viewpoints,joints?))?)
     |(viewpoints,((joints,segments?)|(segments,joints?))?) )? )
     |(viewpoints, ((joints,((segments,sites?)|(sites,segments?))?)
     |(segments,((joints,sites?)|(sites,joints?))?)
     | sites,((segments,joints?)|(joints,segments?))?) )? ) )? )

<!ATTLIST Humanoid
      bboxCenter        %SFVec3f;   "0 0 0"
      bboxSize          %SFVec3f;   "-1 -1 -1"
      center                %SFVec3f;   "0 0 0"
      info              %MFString;  #IMPLIED
      name              %SFString;  #IMPLIED
      rotation          %SFRotation;      "0 0 1 0"
      scale             %SFVec3f;   "1 1 1"
      scaleOrientation  %SFRotation;      "0 0 1 0"
      translation       %SFVec3f;   "0 0 0"
      version           %SFString;  #FIXED      "1.1"
      nodeTypeHint      NMTOKEN           #FIXED      "Transform"
      DEF         ID          #IMPLIED
      USE         IDREF       #IMPLIED>
```

Figure 32. Fragment from HumanoidAnimation.dtd. From (X3D, 2000).

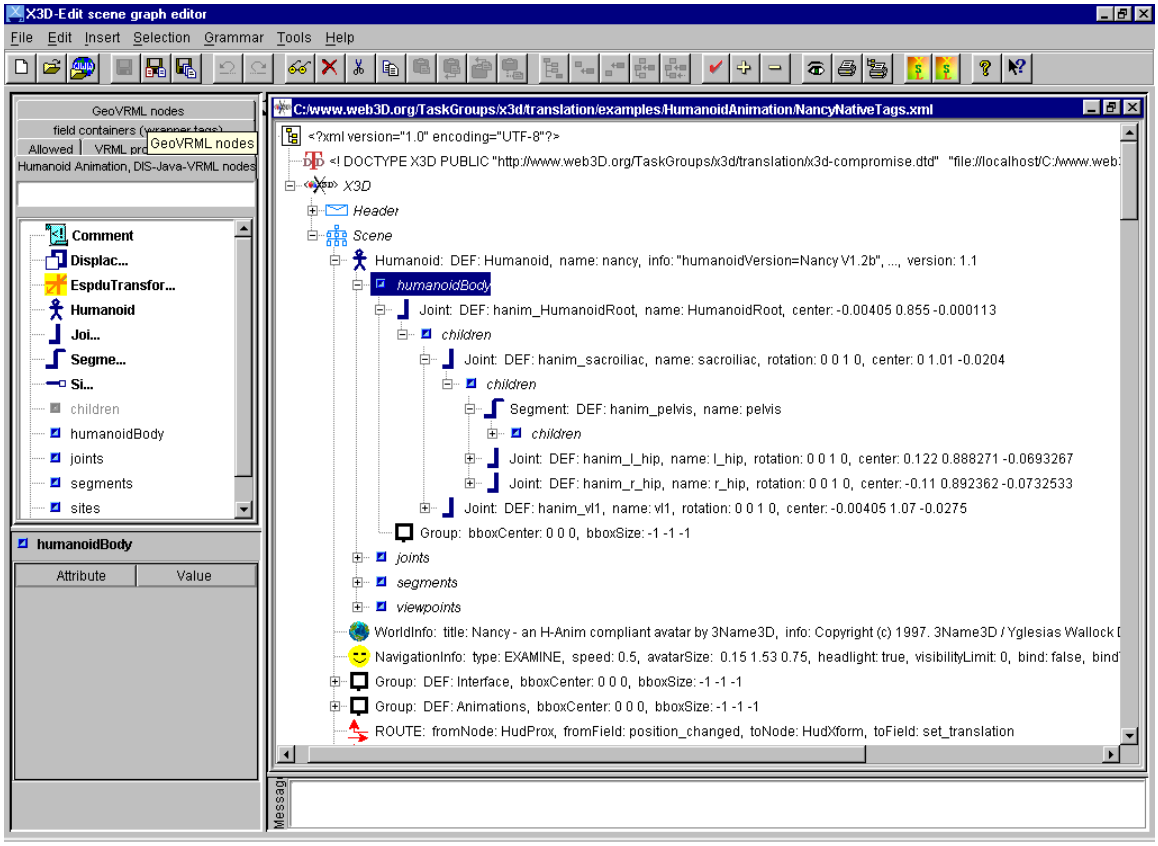


Figure 33. Screen Shot of the NancyNativeTags.xml Showing the H-Anim Native Tag Set in the Leftmost Panel. Also Note the EspduTransform Native Tag Also in the Leftmost Panel.

2. XSL Script Changes

Changes to the XSL Stylesheet had to be implemented in order to format and render H-Anim content from its new native tag set. These changes are merely additions to the existing document, allowing the H-Anim extension to be treated as if it were a VRML97 Specification node. Figure 34, a code fragment from the updated X3DtoVRML97.xsl, illustrates the format of the EXTERNPROTO of the Joint node that is written into a translated H-Anim VRML file, thus allowing it to be rendered by a VRML browser.

```
<xsl:text>]</xsl:text>
  </xsl:if>
    - <xsl:if test="//Joint and
        not(//ExternProtoDeclare[@name='Joint'])">
        <xsl:text>EXTERNPROTO Joint [</xsl:text>
        <xsl:text>exposedField SFVec3f center # 0 0 0</xsl:text>
        <xsl:text>exposedField MFNode children # []</xsl:text>
        <xsl:text>exposedField MFFloat llimit # []</xsl:text>
        <xsl:text>exposedField SFRotation
           limitOrientation # 0 0 1 0</xsl:text>
        <xsl:text>exposedField SFString name #  ""</xsl:text>
        <xsl:text>exposedField SFRotation
           rotation # 0 0 1 0</xsl:text>
        <xsl:text>exposedField SFVec3f scale # 1 1 1</xsl:text>
        <xsl:text>exposedField SFRotation
           scaleOrientation # 0 0 1 0</xsl:text>
        <xsl:text>exposedField MFFloat
           stiffness # [ 0 0 0 ]</xsl:text>
        <xsl:text>exposedField SFVec3f
           translation # 0 0 0</xsl:text>
        <xsl:text>exposedField MFFloat ulimit # []</xsl:text>
        <xsl:text>] [</xsl:text>
          <xsl:text>"NancyProtosTranslated.wrl#Joint"</xsl:text>
       <xsl:text>"C:/www.web3D.org/TaskGroups/x3d/translation/
          examples/HumanoidAnimation/
          NancyProtosTranslated.wrl#Joint"</xsl:text>
```

Figure 34. A Code Fragment from the Updated X3DtoVRML97.xsl showing the formatting for the EXTERNPROTO declaration of Joint. From (X3D, 2000).

3. Lessons Learned

The implementation of the H-Anim native tag set in the X3D-Edit authoring tool considerably speeds development work and debugging on existing H-Anim content. The use of the native tags eliminates costly time dealing with PROTO/ EXTERNPROTO declarations and instantiations while the H-Anim DTD allows work to be validated during development providing early identification of errors. While no H-Anim content has been developed from scratch with the native tag set, the authoring tool will provides users the same (if not greater) benefit when compared to extension work done on existing

content. A major lesson learned through the implementation of the H-Anim tag set was that CosmoPlayer does not support the EXTERNPROTO model mandated in the VRML97 Specification well enough to render H-Anim native tag set VRML translations, although the other major VRML browsers can. We suspect this bug prevented the development of a generalized H-Anim EXTERNPROTO specification. This unfortunately limits development for DIS-Java-VRML inclusion of native tag XML H-Anim content since the other browsers do not yet support the security model.

E. SUMMARY

This chapter discussed the reasons for converting to X3D and the experience of converting a H-Anim compliant human from VRML to XML and then translating back to VRML. Subsequently, the development and implementation of a native tag set in X3D for the H-Anim 1.1 Spec is discussed.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. MOUNTING OF HUMAN ENTITIES

A. INTRODUCTION

This chapter discusses the mounting and aggregating of the human entities to non-human entities. Entities transmitting DIS protocol Entity State PDUs, expressed in relative coordinate systems versus a world coordinate system, while the entities maintain their true real-world location, are addressed. A basic mounting algorithm is defined and its usage is illustrated through its implementation in the entities' action interpreters and the VRML scene. Finally, the topics of collision, rotations in a relative coordinate system and network latency are discussed in terms of the implementation.

B. LOCAL COORDINATES IN DIS

The DIS protocol is designed to pass all position information in terms of world coordinates. However, in order to aggregate virtual humans, the use of local coordinates was required. Otherwise, the high precision relative motion needed for team activities is not possible across network delays or in georeferenced locations. A non-physical entity, called an aggregation entity, was created to act as the centroid position from which to determine the relative coordinates of mounted or aggregated entities. As a non-physical entity, it lacks the ability to perform actions directly effecting the CTF game play, i.e. it cannot fire, collide with another entity, be destroyed or capture the flag. However, it maintains CTF's physically based modeling framework of movement whether it is directing virtual humans representing DI or a virtual helicopter. The aggregation entity

can present an iconic representation during game play as a reference to the human controller as shown in Figure 35, or otherwise set to be invisible. Even though the aggregation entity is a non-physical entity, it provides the control and coordination of the group actions of the physical entities aggregated with it.

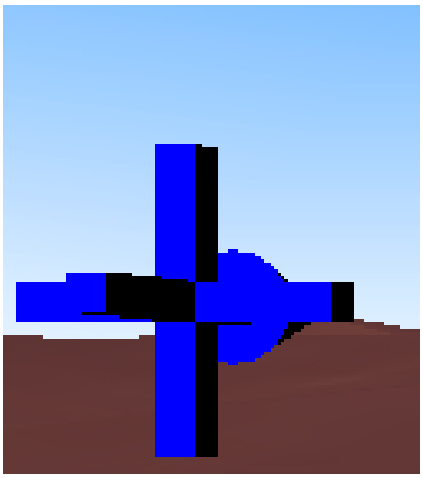


Figure 35. The Aggregation Entity: Directional Three-Axis Icon.

C. MOUNTING HUMANS

The vast majority of entities in a VE are controlled singly and independently. This limits the ability to perform moves requiring any team effort or coordination. Coordination of group activity and movement is usually the responsibility of their "real world" human controllers. This "real world" coordination requirement makes group operations difficult to impossible. By aggregating entities into groups or mounting them on non-human entities, realistic group behaviors can be performed and exhibited.

In the manner that DIS is designed and normally implemented, entities' positions are presented in a shared world coordinate system referenced to a terrain database. However, in this thesis research, mounted entities use relative coordinates in relation to

the world coordinate position provided by the aggregation entity. The use of relative coordinates in networked VEs is usually problematic, since the networked ghost entities normally interpret the relative position information as world coordinates, subsequently rendering the entity in the wrong location. This problem was addressed through the use of articulated parameters to the ESPDU, to denote whether an entity was mounted to the aggregation entity, in both the mounted entity and the aggregation entity. The ESPDUs are sent across the network and the ghost entities are informed of their mounted status. In this implementation, a true position ESPDU is maintained in the master entity only. The true-position ESPDU is used, as a data structure to maintain entity state data that otherwise is lost through the conversion to relative coordinates. This facilitates switching from mounted state, collision and weapons firing, and is critical to the accurate movement of mounted entities. Individual behaviors are determined by the magnitude of linear velocity, and while aggregated they perform their behaviors based on the magnitude of the linear velocity vector, to an upper limit, formed by summing the linear velocities of the aggregation entity and mounted entity. This provides appropriate behaviors unless the velocity vectors are in opposite directions.

The mounting algorithm used in this research essentially replicates vector addition of the velocity vectors of the mounted entity and the aggregation entity. The basic steps to the algorithm are as follows: move the mounted entity as if it were an independent entity, determine a relative vector from the mounted entity to the aggregation entity, move the aggregation element, add the relative vector to the aggregation entity's new position

to determine the mounted entity's final position, and of course, repeat until disaggregation. This basic algorithm is illustrated in Figure 36.

1. Implementing the VRML Scene

The decision on the final form of this research's implementation came after almost a year of test and development work involving a variety of networking and rendering implementations. The evaluation criteria dealt with reducing network latency, maintaining PDU send rate at a level that would not cripple further scalability, ease of implementation and reduction of complexity to achieve the highest possible frame rate in a complex maneuver scene. In its most complex behaviors, such as mounting the helicopter, a average frame rate of approximately four frames per second is maintained on a 512 MB RAM machine, dropping from an average of eight to nine frames per second on less complex behaviors. In the end, two implementations were fully developed to handle a mix of five human entities and single aggregation entity. Only the final implementation developed includes a helicopter entity and above ground-level maneuver.

The first of the fully developed implementations involved the switching of inlined files containing the same entity but presenting it as either a mounted or unmounted entity. A scene graph fragment of this implementation is shown in Figure 37. The decision to render the entity as mounted or unmounted was provided from the mounted state articulated parameter from the aggregation entity, which was routed to an ECMAScript internal to the scene graph for implementation, as illustrated by Route1 in Figure 37.

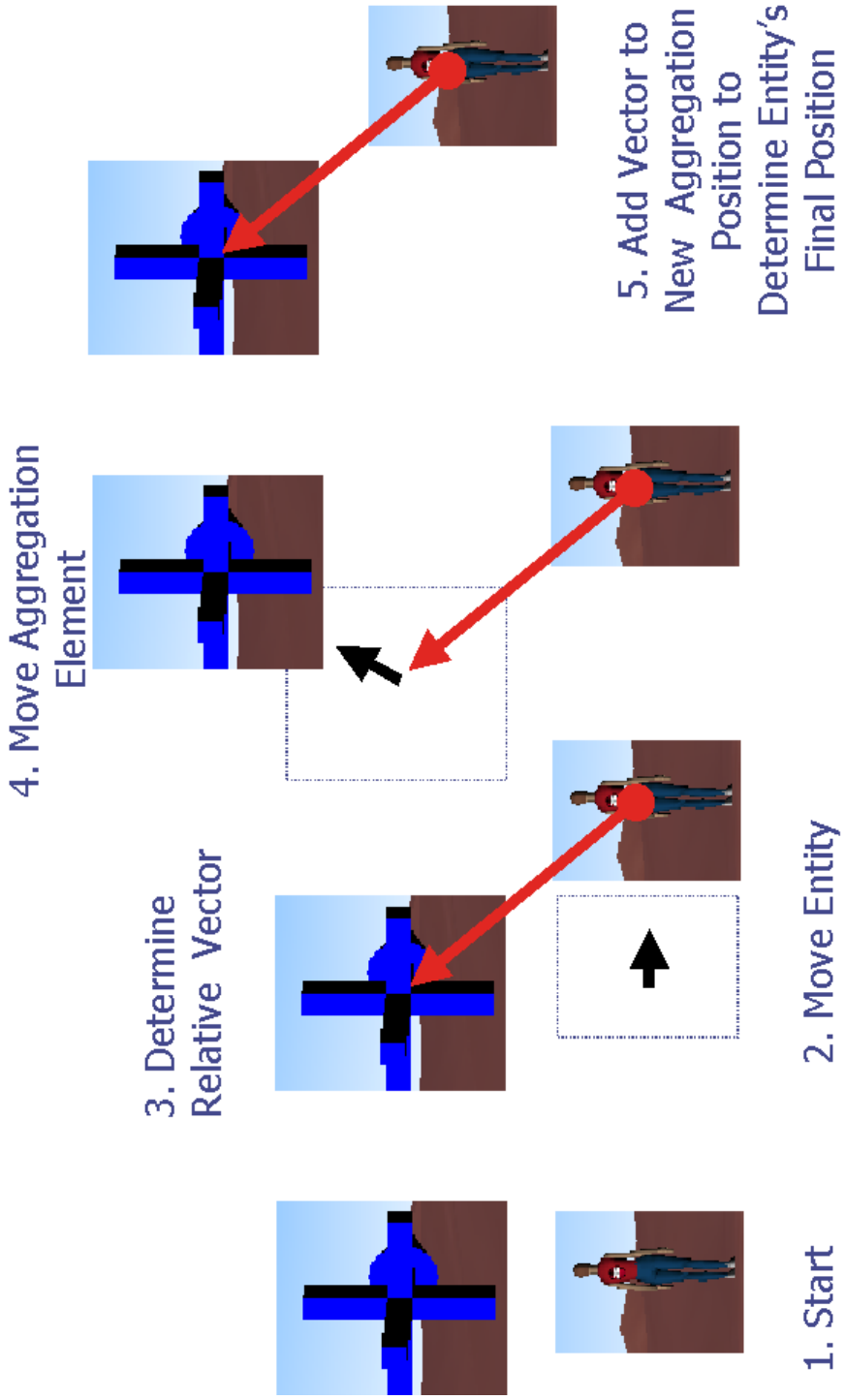


Figure 36. Mounting algorithm illustrated. A moving humanoid first determines vector distance and velocity differences with the aggregation vector, then adjusts to match.

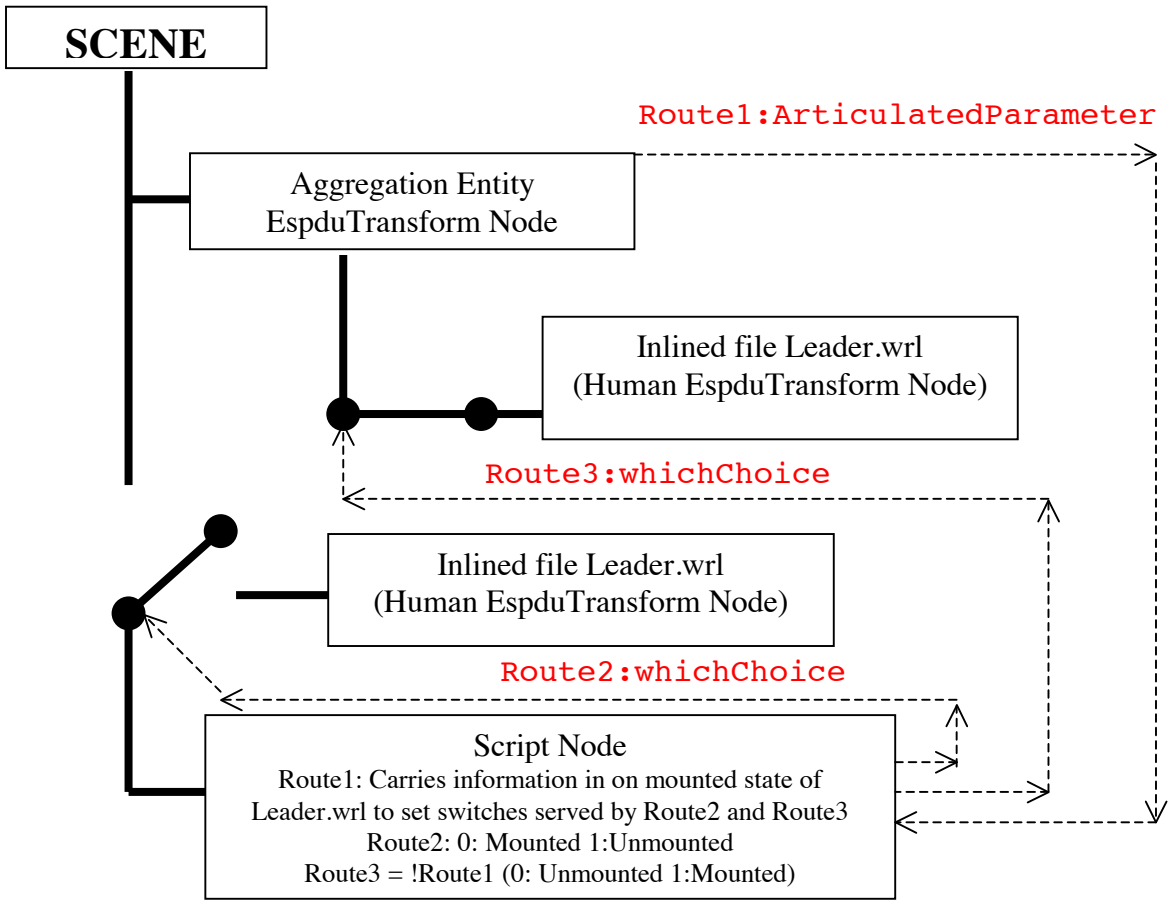


Figure 37. Fragment of a VRML scene graph showing the first fully developed test implementation dealing with switching the human entity between a scene graph child and a scene graph peer of the aggregation entity.

The Script node then routed the implementation to the Switch nodes, as shown by Route2 and Route3 in Figure 37, which in turn rendered the human entity to the correct position in the correct mounted state. Unfortunately, there were two problems which prevented this test implementation from being the final product. The first problem dealt with the switching node, which is required according to the VRML specification to pass all information along its routes to every choice in a Switch node. This led to the artifact, due to DIS and even though one Espdu was transmitted across the network, that the human

entity was rendered in both its mounted position relative to the aggregation entity and to its world coordinate position when the entity was in the mounted state. While a work around solution of a higher-level Switch node over the unmounted inline file was implemented to set the world coordinate human entity non-visible when in the mounted mode, this was an unacceptable fix since its implementation affected future scalability. The other element dealt with the loss of the active viewpoint when switching from a mounted to a nonmounted state and vice versa. This was caused by reloading the inlined file with the Switch node at the change of state. The active viewpoint was instead bound to the viewpoint prior to the currently active one prior to state change. This viewpoint-hysteresis problem was also felt to be unacceptable in terms of usability and acceptance of the model.

The final implementation involves sophisticated addition and deletion of VRML routes at run-time to provide the correct implementation and rendering of an entity either in relative coordinates (mounted state) or in world coordinates (unmounted state). Scene graph fragments of the final implementation are shown in Figures 38, 39, 40 and 41. The implementation involves the aggregation entity's EspduTransform node as a scene graph peer of a Transform node wrapped around an inlined file containing the EspduTransform node of an entity that can execute mounting. Like the first fully developed implementation, the ESPDUs from the action interpreter classes transmit to the EspduTransform script, which subsequently passes them to the VRML file as seen in Figure 38. The aggregation entity then routes the mounted state articulated parameter for

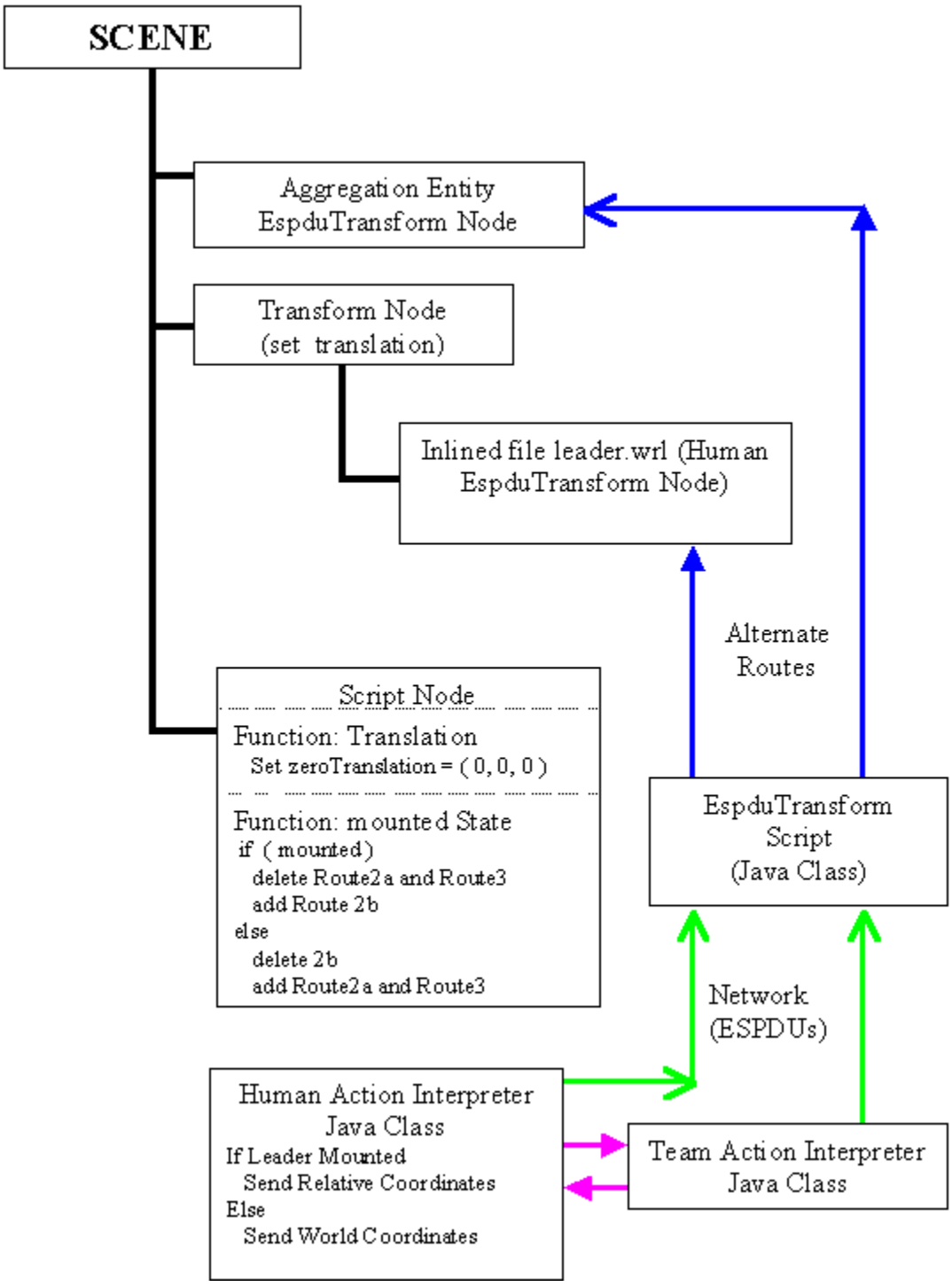


Figure 38. Java Classes updating the VRML scene through ESPDUs via the EspduTransform Java (language) Script class.

the specific entity involved to a ECMAScript Script node internal to the VRML file, as shown in Figure 39. Based of the articulation parameter's value, the logic resident in the

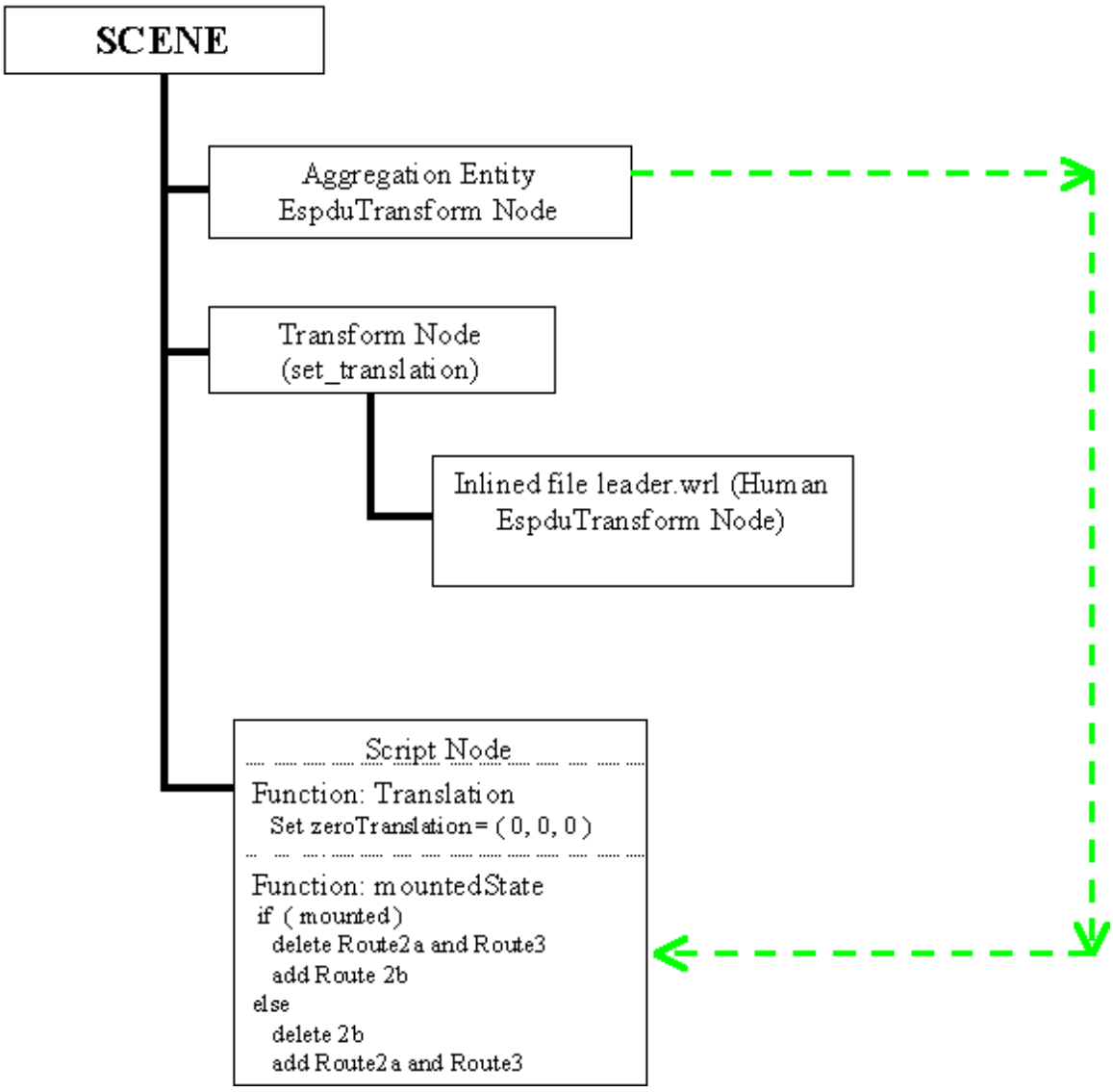


Figure 39. The mountedState ECMAScript receiving the MountedState articulated parameter value via ROUTE from the VRML EspduTransform Node.

entity ECMAScript calls the browser in order to delete and then add the designated routes, which allows the correct passing of translation values to be added to the top of the

entity's matrix stack. If the entity is in a mounted state, the position of the aggregation entity is routed to the Transform wrapper node, as shown in Figure 40, following the deletion of ROUTEs 2a and 3, and the addition of ROUTE2b. If the entity is in a

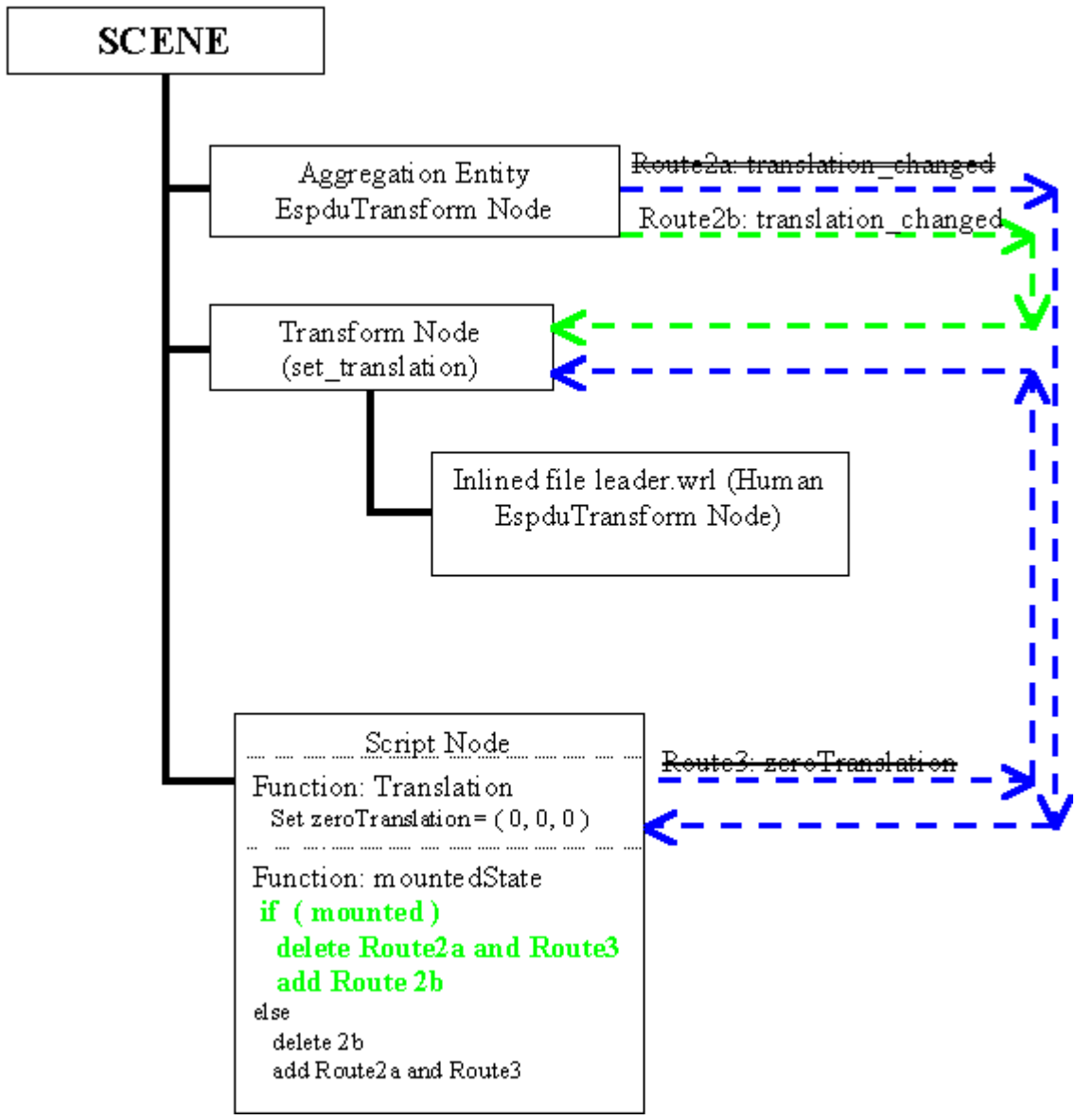


Figure 40. Illustrating the Add and Delete ROUTEs when the entity is Mounted.

nonmounted State, then the translation from the aggregation entity is routed to the Script node, it is zeroed out and then the zeroed translation is routed to the Transform wrapper node, as shown in Figure 41, after ROUTE 2b is deleted and ROUTEs 2a and 3 are added.

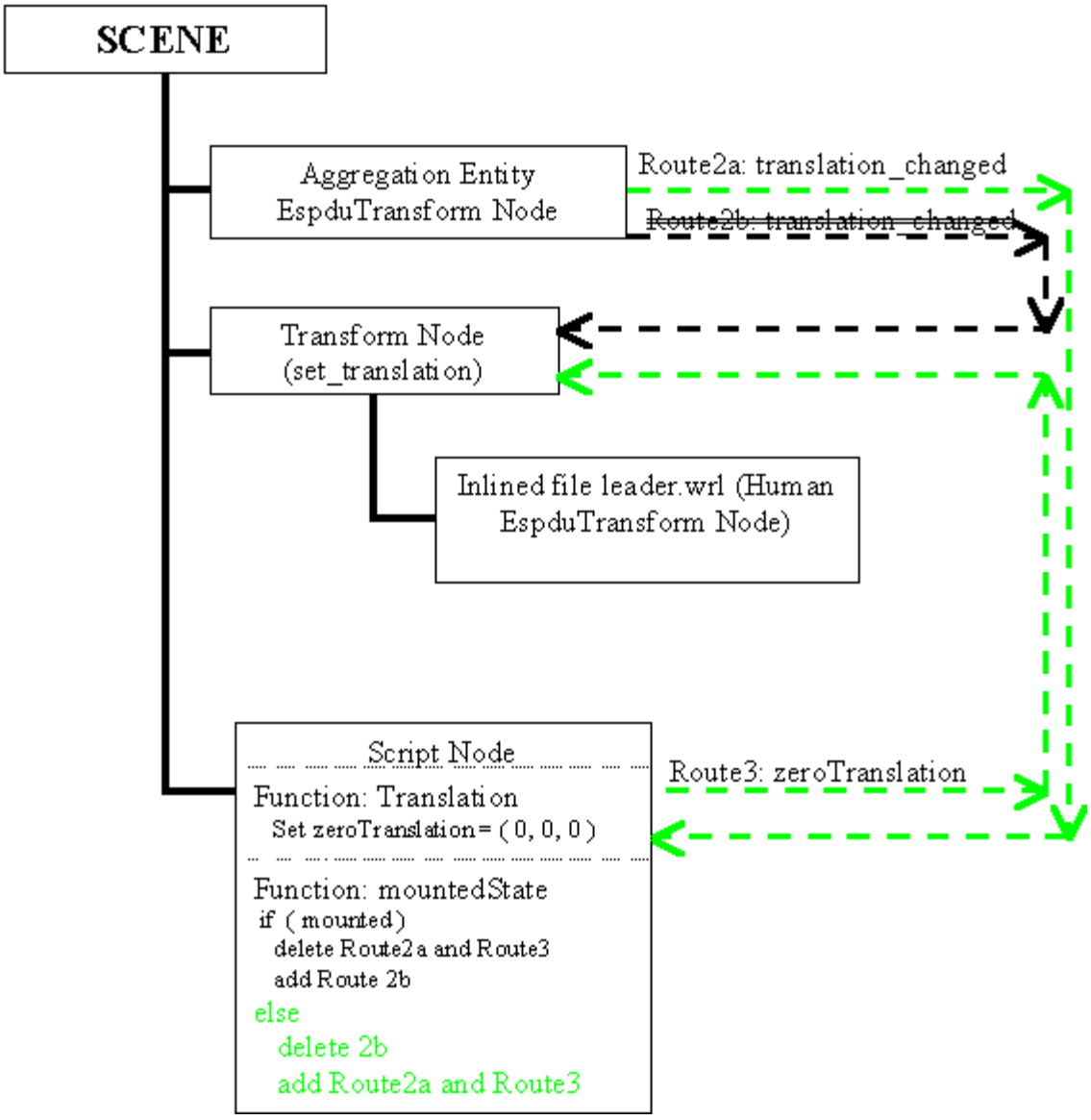


Figure 41. Illustrating the Add and Delete Routes When the Entity is Unmounted in the Final Implementation.

The ECMAScript function mountedState which sends the calls to the browser to add and delete the routes, is shown in its entirety in Figure 42. The boolean flags routeUnloaded and routeLoaded are activated by passing the mounted state articulation parameter value from the aggregation entity. In the VRML rendering of the scene,

```
function mountedState ( mState, timeStamp ) {
    print ('Leader mountedState =' + mState); // 0=unmounted, 1=mounted
    if (mState == 0)  {
       if (!routeUnloaded)  {
          Browser.deleteRoute ( fromNode, 'translation_changed',
             thisNode, 'translation' );
          Browser.deleteRoute ( thisNode, 'zeroTranslation',
             toNode, 'set_translation' );
          routeUnloaded = true;
          print ('Just deleted Local Translation Routes. routeUnloaded = '
                 + routeUnloaded);
       }  // end if
       if ( !routeLoaded )  {
           Browser.addRoute ( fromNode, 'translation_changed',
              toNode, 'set_translation' );
           routeLoaded = true;
           print ('Just added Espdu Translation Routes. routeLoaded ='
                  + routeLoaded);
       }  // end if
    }  //  end if
    else  {
       if ( routeLoaded )  {
       Browser.deleteRoute ( fromNode, 'translation_changed',
          toNode, 'set_translation' );
       routeLoaded = false;
       print ('Just deleted Espdu Translation Routes. routeLoaded = '
              + routeLoaded);
       }  //  end if
       if (routeUnloaded) {
          Browser.addRoute ( fromNode, 'translation_changed',
             thisNode, 'translation' );
          Browser.addRoute ( thisNode, 'zeroTranslation',
             toNode, 'set_translation' );
          routeUnloaded = false;
          print ('Just added Local Translation Routes. routeUnloaded = '
                 + routeUnloaded);
       }  //  end if
    }  //  end else
}  //  end mountedState
```

Figure 42. From the file the NancyTeamAddRoutes.wrl, ECMAScript Function mountedState which dynamically adds and deletes VRML ROUTEs at run-time.

mounted and unmounted entities can be visually identified through the color of their trace values. See Figure 43. If the entity is mounted, the trace color represents the team color

(in this case the blue team), but if the entity is unmounted, though normally on the blue team, the trace color is green.

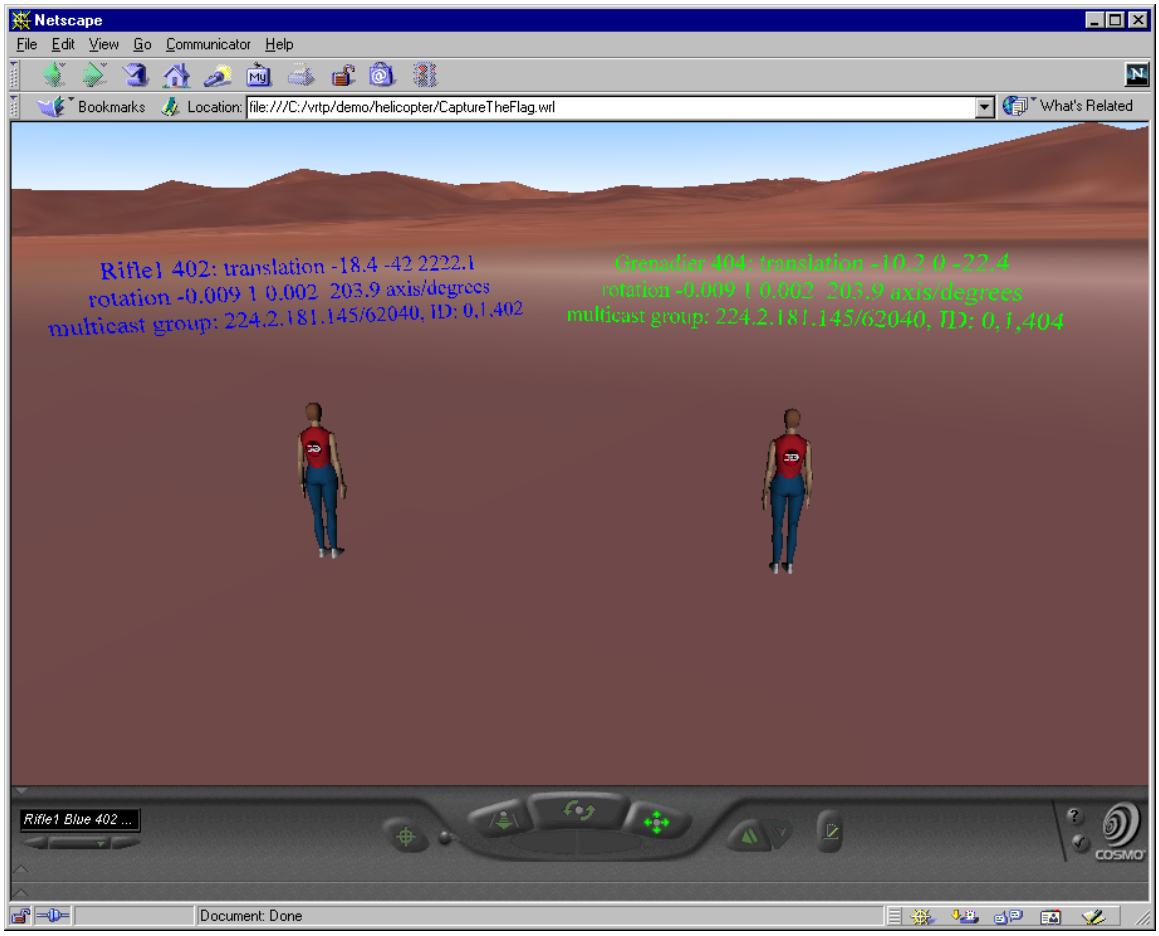


Figure 43. Demonstrating Mounted (Blue) and Unmounted (Green) trace colors.

2. Dealing with Collision

While collision while in the mounted state originally appeared to be problematic, in fact most issues were easily solved. First, the aggregation entity itself is implemented

as a non-physical entity with the ability to have a visible icon assist the user. Therefore, the aggregation entity only deals with collision in regard to the ground plane in order not to place its mounted entities below the world surface. The mounted entities themselves currently have their collision with other mounted team entities disabled. This allows them to mount the helicopter for air assault operations, since group movement under the aggregation entity is neither optimized nor prioritized. Future work can add greater realism to the simulation. The mounted entities do still deal with ground collision, fire, detonation and non-mounted entities, primarily using the stored true position ESPDU held only in the master entity.

3. Dealing with Rotation

In the original implementation testing and development, the mounted entities were provided with a relative position as well as an orientation. Unfortunately, this caused a further matrix stack transformation within the relative coordinates requiring an additional transformation to allow the mounted entity to be rendered in its true position as shown in Figure 44. While this was manageable dealing with only static entities, the complexity of the ongoing calculations combined with the dead-reckoning algorithms in the EspduTransform Java (language) Script class file and network latency led to unsatisfactory frame rate and marginal accuracy in positioning. Therefore, the final implementation had the mounted entities maintaining independent heading and orientation state information except under the special circumstances of the human entities unloading the helicopter or the helicopter itself under mounted flight conditions. These special cases were handled through a direct feed of the aggregation entity's

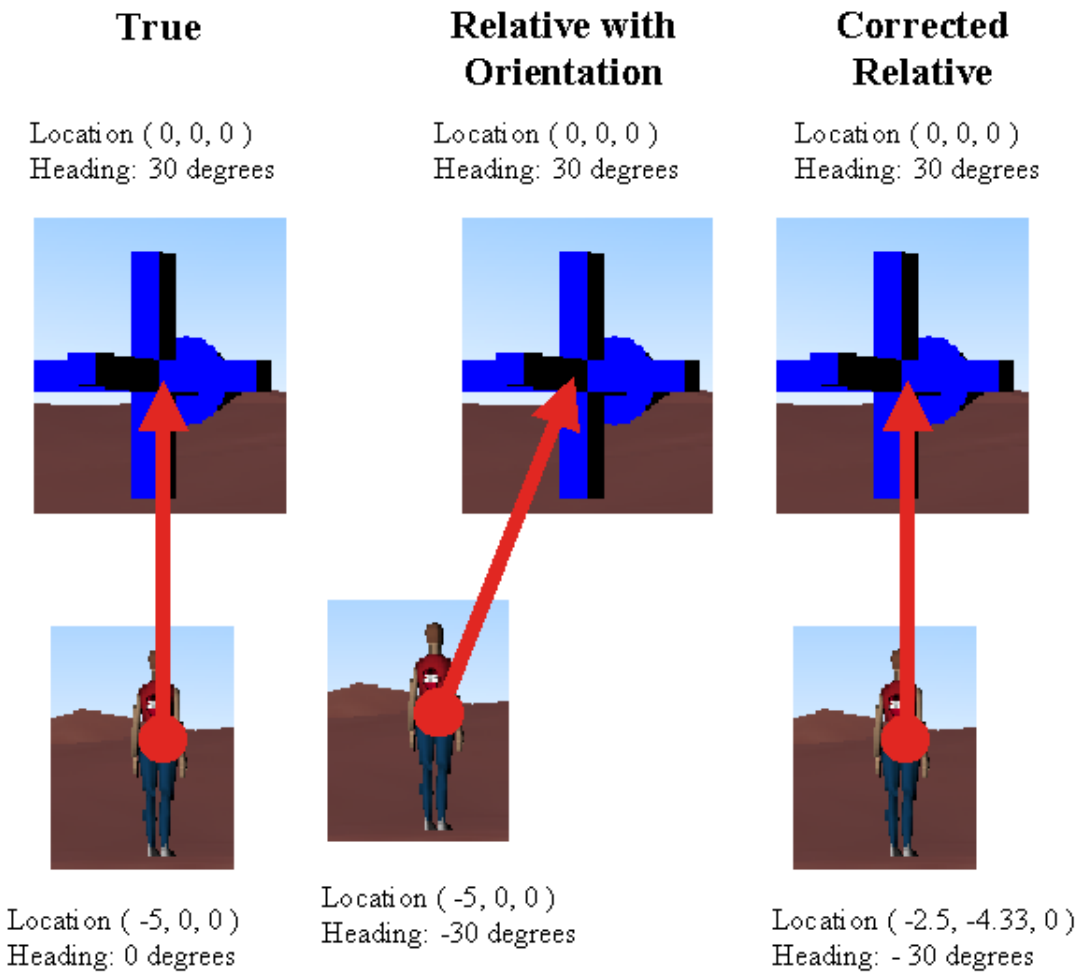


Figure 44. Transforming Relative Coordinates with Orientation Back to the True Position.

heading information to the mounted entity, controlled by boolean flags within the mounted entity's action interpreter.

4. Dealing with Network Latency and PDU Send Rate

The final scene graph implementation was chosen partially for its ability to minimize network latency effects while not executing extremely fast PDU send rates. The aggregation entity transmits PDUs at an average of slightly higher than ten per

second (no less than every 100 milliseconds), the human entities transmit PDUs at an average of two per second (every 500 milliseconds) and the helicopter entity transmits PDUs at slightly higher than two per second (no less than every 500 milliseconds). For the most part, these send rates perform adequately for the demonstrations. The implementation proves to be problematic in situations where entities are very close to the aggregation element or when the aggregation element makes sudden shifts in orientation or linear velocity. This feature became readily apparent during implementation work of the entity-bound and dismount-helicopter behaviors. From these experiences, a hold behavior, which holds a mounted entity in its true location regardless of the motion of the aggregation entity, was developed as a tracking improvement. However, even with the addition of this behavior, problems are still encountered involving mounted entities in close proximity to the aggregation element. While increasing the transmission rates of the mounted entities may further minimize this undesired artifact, it is an inappropriate long-term solution because it minimizes future scalability of the VE by unnecessarily taxing network bandwidth, available computing power and rendering power of readily available personal computing systems. A predictive-behavior aggregation state PDU is worth considering as a further development and implementation task to further optimize such challenging behaviors in future work.

D. SUMMARY

This chapter discusses the mounting or aggregating and disaggregating of the human entities to non-human entities, primarily an aggregation entity. Linking an

aggregation entity to a moving vehicle is also examined. Issues involved with entities transmitting DIS protocol entity state PDUs expressed in relative coordinate systems, while maintaining their true world location are addressed. The definition and implementation of a basic mounting algorithm is presented and its use is illustrated through its implementation in the entities' action interpreters and the VRML scene. Collision, rotations in a relative coordinate system and network latency are also discussed in terms of the final implementation results.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. MULTIPLE-ENTITY, SINGLE-USER INTERFACE

A. INTRODUCTION

This chapter discusses the implementation of the multiple-entity, single-user interface. The use of rule-based movements to define group behaviors in a networked VE is described. Individual behavior implementations to simulate tactical activities are described and illustrated. Supported team behaviors include None, Hold, Column, Line, Wedge, Mount Vehicle and Dismount Vehicle. The master control panel, its components and the underlying action interpreters are also examined in this chapter.

B. RULE-BASED MOVEMENTS AND THE BEHAVIORS

The group behaviors are defined through a series of rules that define a goal position and orientation relative to the aggregation entity. For example, the goal position of the entity closest to the aggregation in the Line behavior is a positive 90 degrees from the current heading of the aggregation element and five meters in distance, with the goal orientation (once it has reached the goal position) to match the aggregation entity's current heading. Each entity has a specific rule that defines its goal position and orientation in each behavior. These simple rules enable the entities to mimic the positions of personnel in a US Army Dismounted Infantry (DI) fire team while performing a variety of tactical activities. Once an entity determines its goal position and orientation, it will incrementally move there. When the goal position and orientation is achieved, the entity correctly maintains its position and orientation by matching the linear

and angular velocity vectors of the aggregation entity until a new behavior becomes active. The specific behaviors are described in detail below.

1. None

The behavior None is generally the default behavior of aggregated entities that have not received a command to perform a behavior. While None is the active behavior, aggregated entities will maintain their relative distances to the aggregation entity and will remain in the final position caused by the last active behavior. However, None also allows the entities to move relative to the aggregation entity under user control. This permits actions that require one or more entities to move away from or within the group, such as a reconnaissance during a tactical movement or a leader being called to the front of a formation.

2. Hold

The behavior Hold is used to prevent aggregated entities from moving in conjunction with the aggregation entity without unmounting them. While Hold is the active behavior, mounted entities do not maintain their relative distances from the aggregation entity, and if stationary, will remain in the same true location even if the aggregation entity is moving. Like the None behavior, the Hold behavior allows the aggregated entity to move relative to the aggregation entity under user control to perform tasks separate from the group. Hold is used in the performance of the compound behaviors Mount Vehicle, Dismount Vehicle (right or left) and Bound. It was a necessary addition to the behavior set to achieve these compound behaviors.

3. Wedge

The wedge formation is the basic tactical movement formation for small units in the US Army. The behavior Wedge, seen in Figure 45, is developed through the use of the goal position and orientation to form and maintain the correct form. The rules for the centered or first tier humanoid define the goal position to be negative five meters in the X direction of the relative (body) coordinates of the aggregation element and the goal orientation to match the current heading of the aggregation element. The rules for the second tier of humanoids is more complex. Their goal position is negative ten meters in the X direction and plus or minus (±) five meters (depending on the entity) in the Y direction of the relative coordinates of the aggregation element. In polar coordinates, this translates to an offset distance of approximately 11 meters and an offset angle of around ± 154 degrees (depending on the side the entity occupies in the wedge). In Cartesian world coordinates the goal position of the aggregated entity is calculated as follows:

X component = X component of the aggregation entity's location
+ Cosine (± 154 degrees) * (11 meters) and
Y component = Y component of the aggregation entity's location
+ Sine (± 154 degrees) * (11 meters).

The goal orientation remains the aggregation entity's current heading. The third-tier entities' rules define the goal position as negative 15 meters in the X direction and ± 10 meters in the Y direction of the aggregation entity's body coordinates, and the goal orientation of the current heading of the aggregation entity.

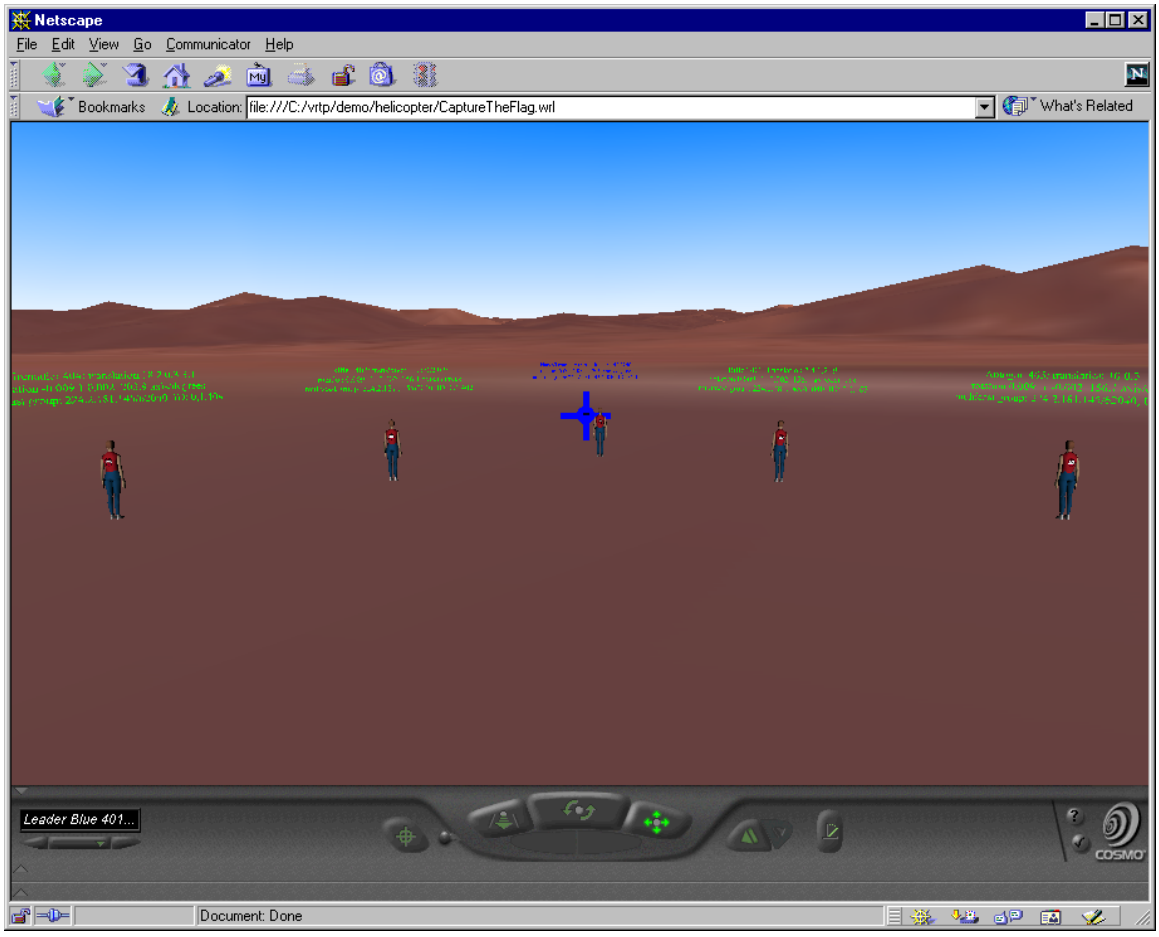


Figure 45. The Nancy Team in the Wedge movement formation.

4. Line

The Line and Bound are the standard tactical formations for an assault. The behavior Line, shown in Figure 46, is easier to define than the wedge. The goal positions of the aggregated entities are negative five, negative ten, -15, -20 and -25 meters in the Y direction of the relative coordinates of the aggregation entity. The goal orientation is the current heading of the aggregation entity.

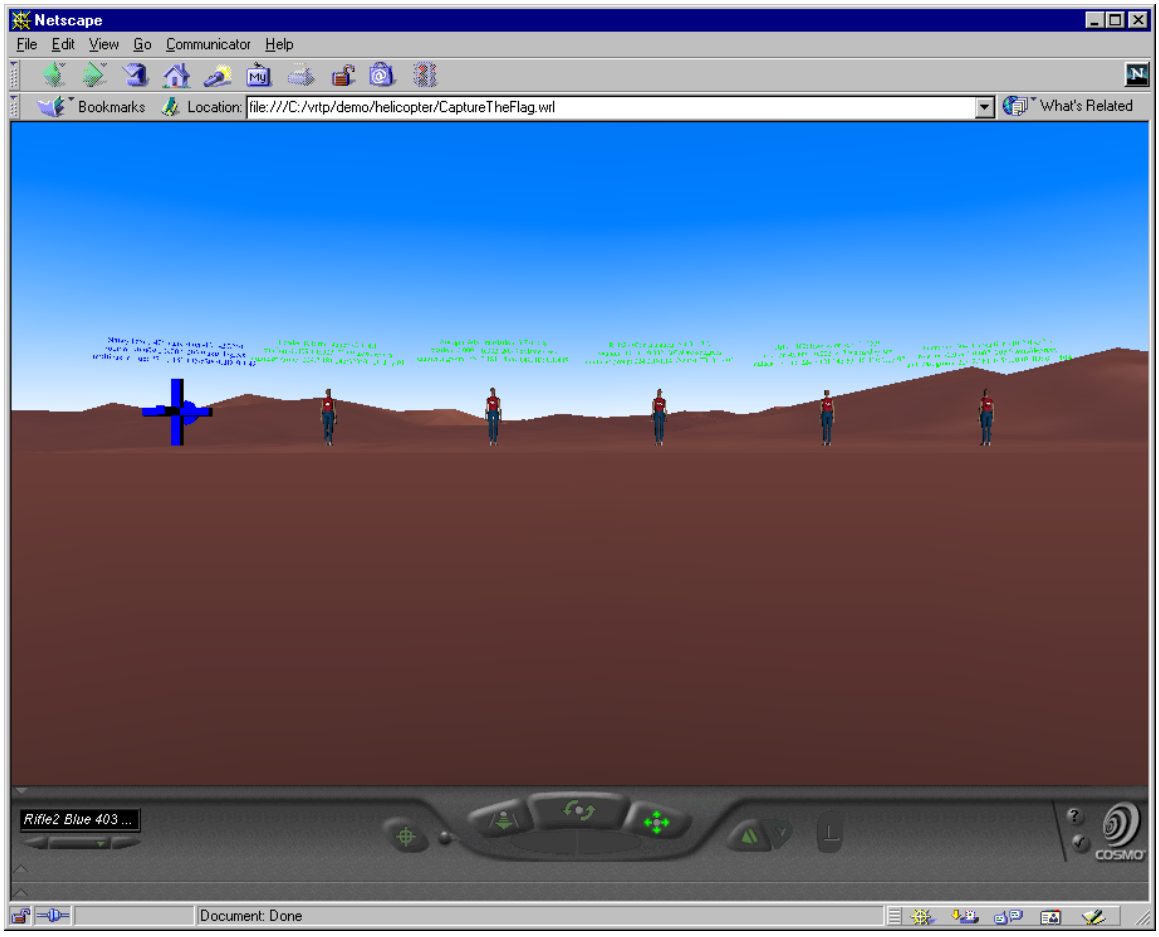


Figure 46. The Nancy Team in the Line movement formation.

5. Column

In the US Army, the column or file is defined in the terms of a modified Wedge. It is used in conditions where command and control of an element becomes difficult when the element is in a more open-movement formation. Its primary usages are when the team is moving in restrictive terrain or during periods of limited visibility. The behavior Column, shown in Figure 47, lies between the Wedge and Line in its ease of definition. The goal positions are defined as positive seven and one half meters, positive

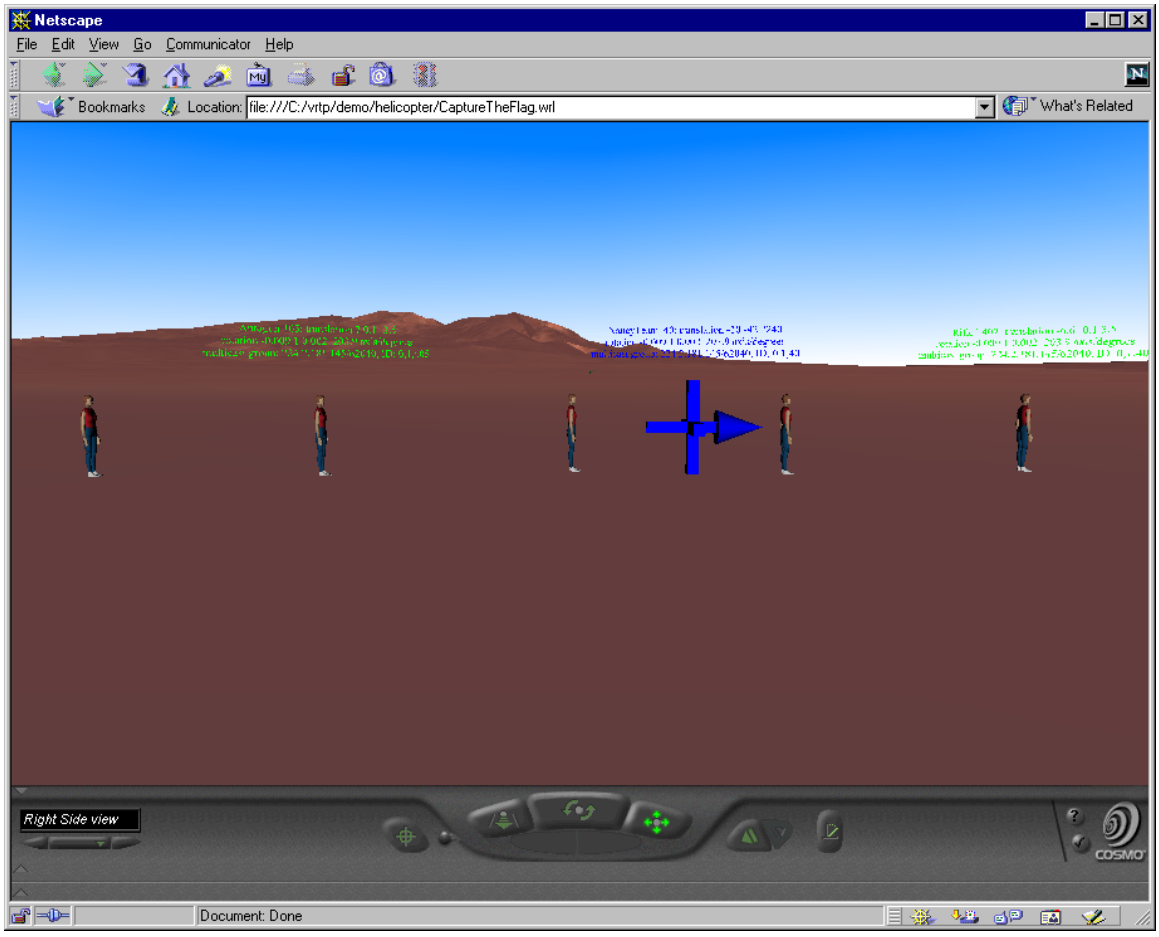


Figure 47. The Nancy Team in the Column movement formation, viewed from the right side.

two and one half meters, negative two and one half meters, negative seven and one half meters and negative 12.5 meters in the X direction in the relative coordinates of the aggregation entity. The goal orientation is the aggregation entity's current heading.

6. Bound

Bound (as in leapfrogging or jumping ahead) is a standard assault movement technique) used in conjunction with the line formation.. The behavior Bound, illustrated in Figure 48, is a compound behavior, combining the line and the hold behaviors.

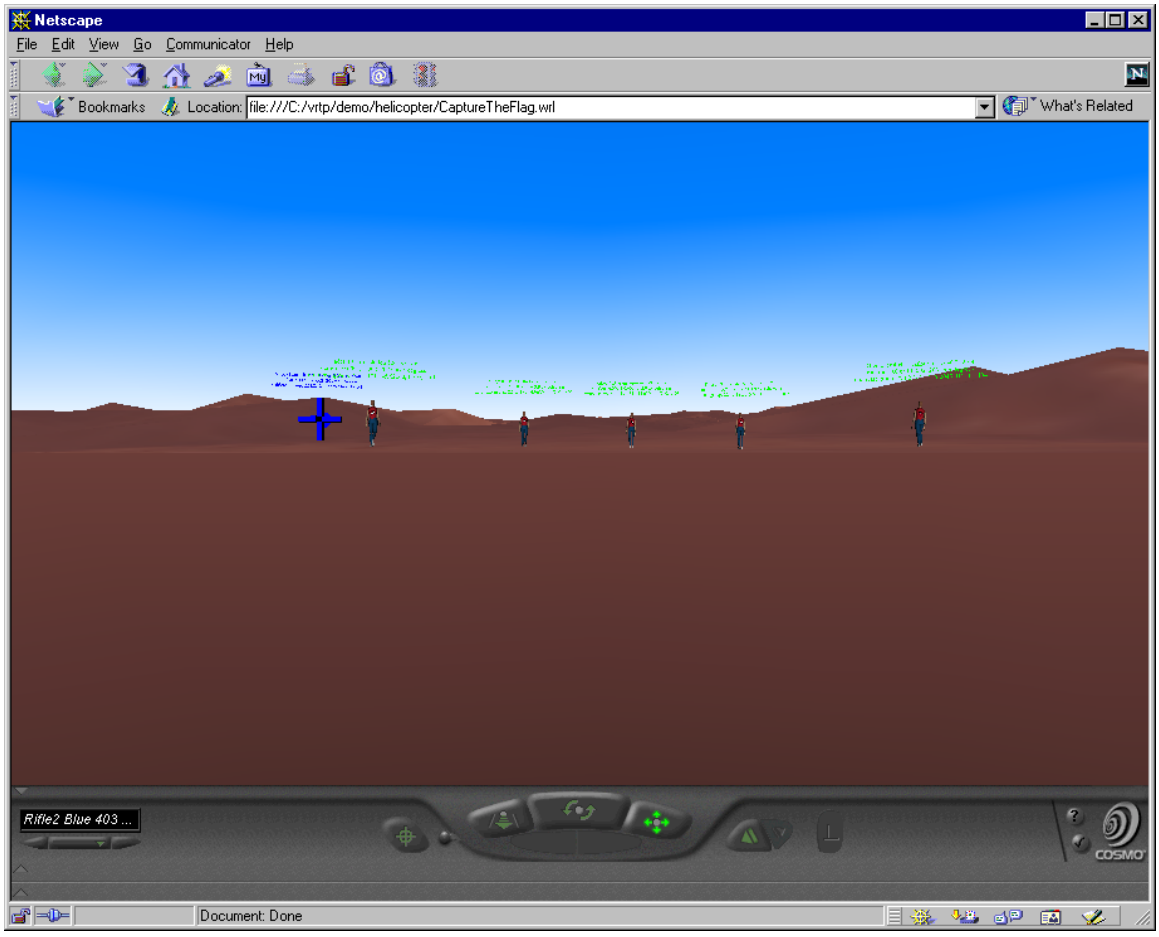


Figure 48. The Nancy Team demonstrating the Bound behavior.

In Figure 48, the two outermost aggregated entities have received hold commands while the other aggregated entities are moving in the line formation. When the aggregation entity receives a command to bound from the user, it issues an order to move the aggregated entities into a line. The aggregation entity then polls the aggregated entities to determine when they are in their goal positions and orientations. Once all aggregated entities have achieved this state, the aggregation entity will order two aggregated entities to hold while advancing in line for approximately 20 meters with the other three entities.

Once the 20 meter distance is covered, the aggregation entity releases the hold order on the two entities with a line command, and begins polling again for the achievement of the line-goal positions and orientations. The previously held entities then advance back into their line positions, and the cycle is repeated until a different movement order is given.

7. Mount Vehicle (Helicopter)

The Mount Vehicle behavior shown in Figure 49 is another compound behavior involving the Line behavior, with additional rules dealing with actually entering the helicopter. At the beginning of the mount vehicle command, the vehicle is ordered to hold. The aggregation entity reads the vehicle's ESPDU and moves to its center and matches its orientation. Simultaneously, the aggregated entities are given an order to go to a line. When the aggregation entity is close to the vehicle's center (within half a meter) and its orientation (within a tolerance of two degrees), it begins to poll the human aggregated entities to ensure they have reached their Line behavior goal positions and orientations. Once they all report that they have reached this state, the team entities turn toward the helicopter by setting their goal orientations to a positive 90 degrees from the aggregation entity's orientation Upon achieving these new goal orientations, they begin to mount the helicopter, actually entering it when they are within a distance of one and one half meters distance from the aggregation entity. Although upon first examination the mounting of the helicopter may seem stilted and formulated, in fact, for safety considerations dealing with the tail rotor area of rotation, main rotor flap and visibility considerations of the pilots and crew chief/door gunners, this mounting behavior closely simulates how helicopters are loaded by teams in the tactical environment.

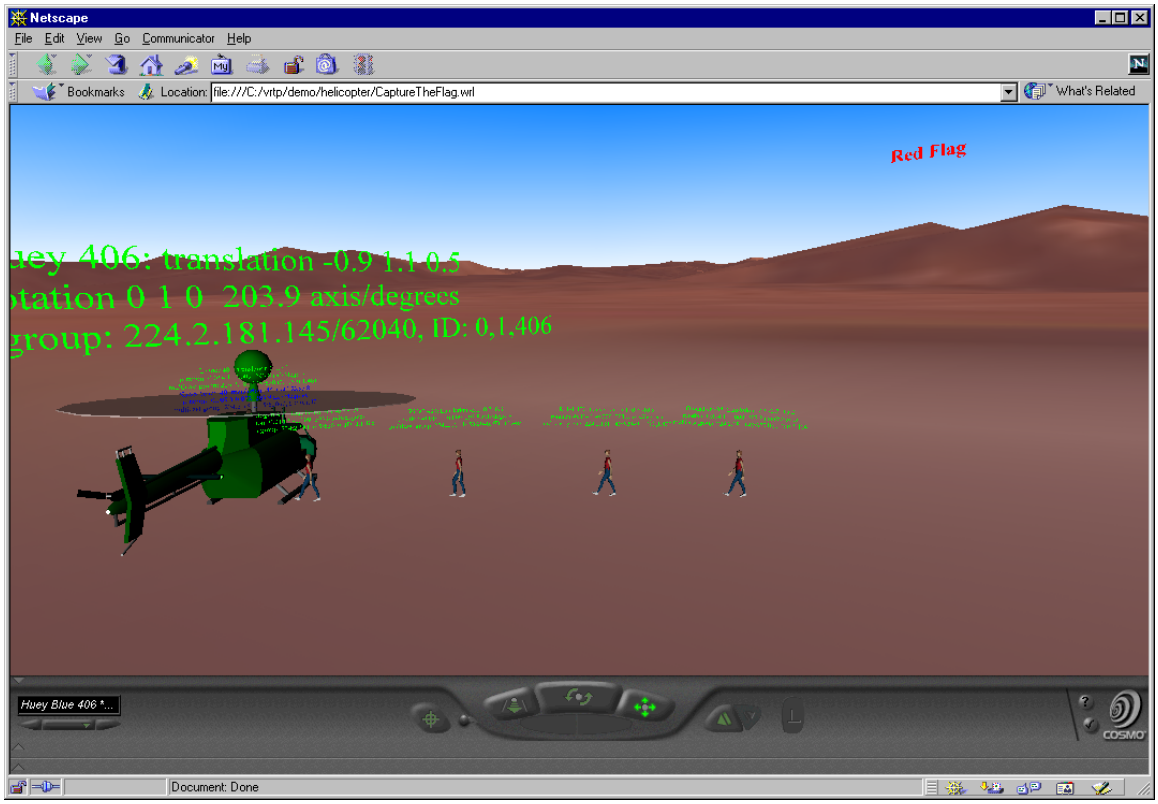


Figure 49. The Nancy Team Loading the Helicopter.

8. Dismount Vehicle Right or Left

The Dismount Vehicle to the right or left behaviors are essentially the same rule set. The initial change of orientation is either ± 90 degrees from the landing heading of the aggregation entity, since during flight conditions the helicopter matches the orientation changes of the aggregation entity. The Dismount Vehicle to the Right is illustrated in Figure 50. The Dismount behavior begins with the aggregation entity orienting to support the correct dismount orientation. Following the orientation change, the first exiting aggregated entity reorients to match the aggregation entity and then begins to move to its goal position. Once it has moved a distance of five meters from the

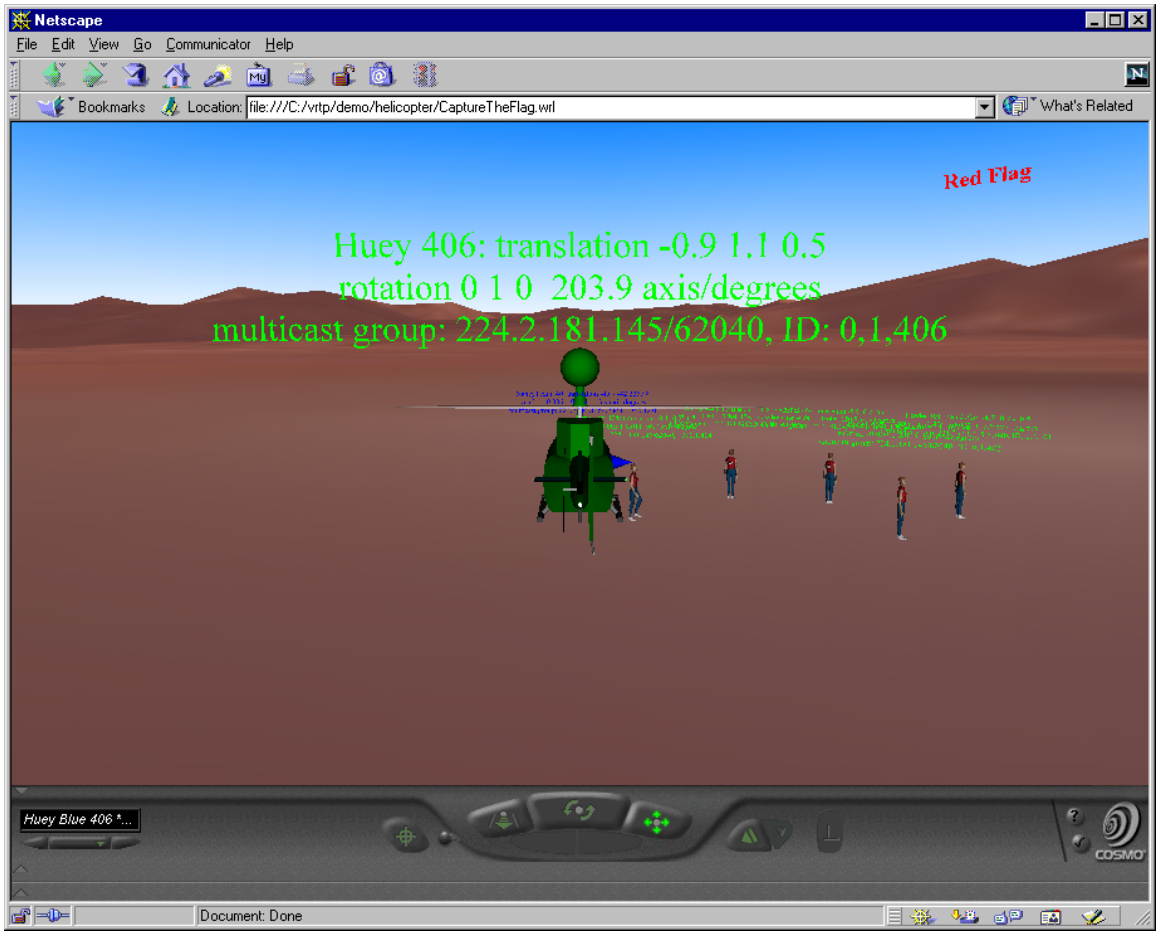


Figure 50. The Nancy Team dismounting the helicopter to the right.

aggregation entity, the team member signals that it has dismounted and the next exiting aggregated entity gets its new rule set to determine its goal position and orientation. When the entities reach their goal position, they are ordered to hold. This allows the aggregation entity to exit the helicopter without beginning motion within the fire team. Of interest, as the team of aggregated entities reach their goal positions, their goal orientations are set so that they face out in a semi-circle rather than maintain the orientation with which they exited. This simulates a hasty defensive position for troops dismounting a helicopter in a tactical situation.

C. THE MULTI-ENTITY SINGLE-USER INTERFACE

The multi-entity single-user interface consists of the master GUI control panel formed by a Java jFrame consisting of jPanels used for the linked control of the aggregation entity and the aggregated human and helicopter entities, and the action interpreters for these elements. The master control panel is shown in Figure 51. Since it is implemented in Java, it executes independently of the VRML scene. This panel is considerably more complex than the single-user, single-entity control panel described in Chapter IV. The top three panels shown in Figure 51 are the controls for the aggregation entities and the commands to be sent to the aggregated entities. The upper left panel represents the basic ground controls of the aggregation entity. Immediately above it is the Other Actions menu which contains items to start and stop the bound behavior. The top-center panel provides button controls for the group behaviors and a visibility button which allows the user to set aggregation entity visible or not. The rightmost panel provides controls for flight operations under the control of the aggregation entity with a higher speed control and a height above the ground climb and descent rate slider. The next five panels hold the human aggregated-entity control panel. An example of this panel is shown in Figure 52. This panel has been redesigned for greater usability and due to the addition of required features for aggregation. Additional features include a

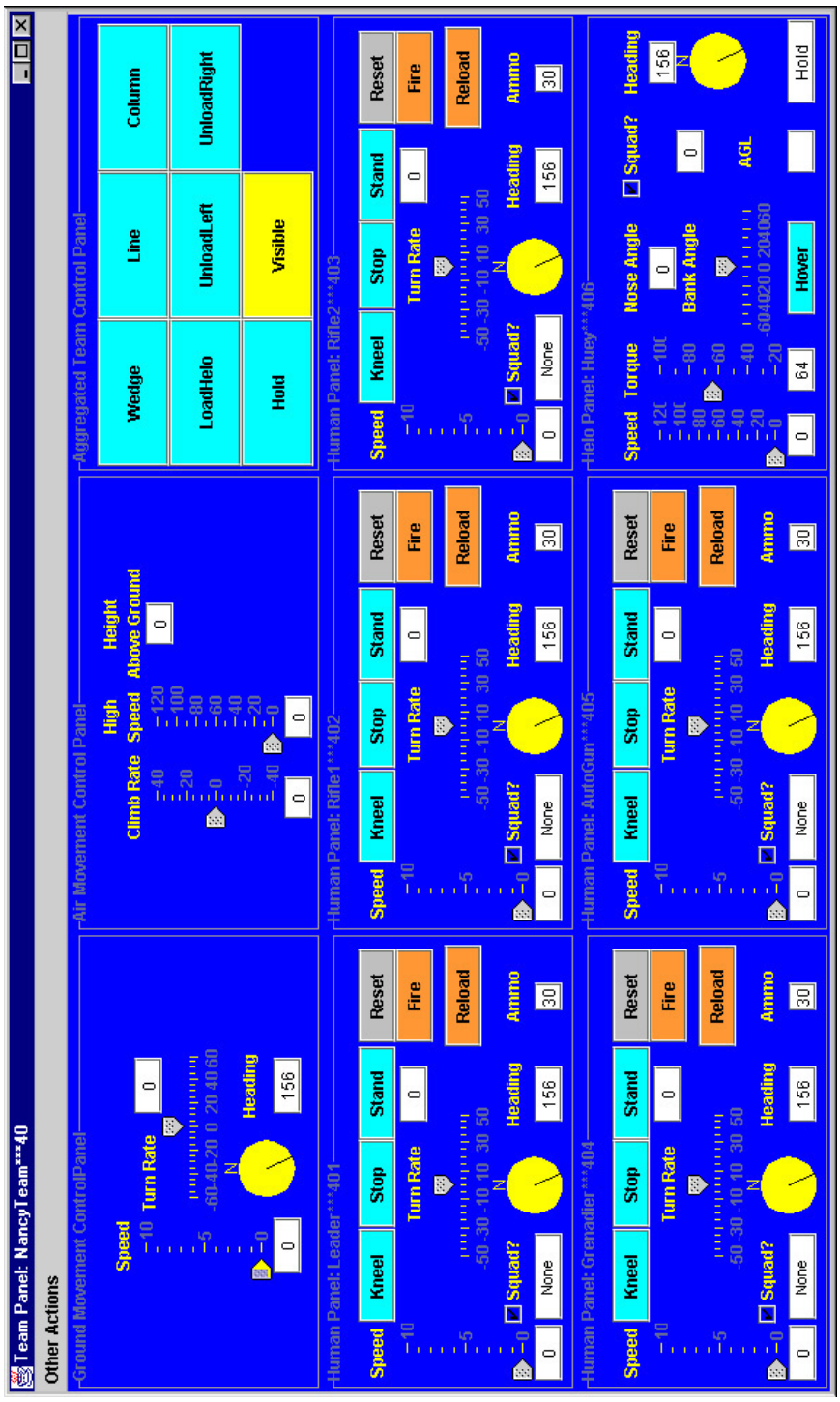


Figure 51. The Multi-Entity Single User GUI team control panel. The top row of subpanels controls the aggregation entity movement with buttons to trigger the behaviors for the team. The second two rows of subpanels control individual humans and the helicopter, either in relative or independent world coordinates.

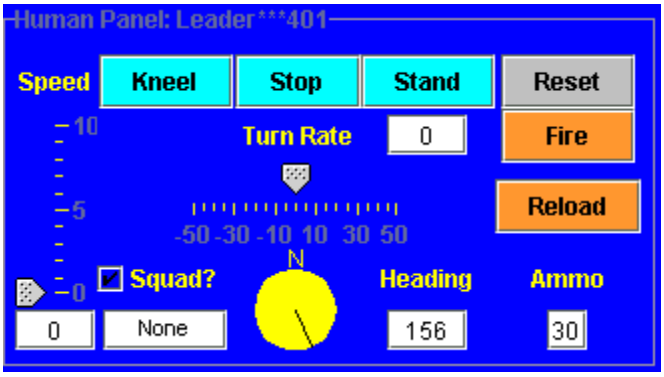


Figure 52. The Human Panel subcomponent to the team control panel.

heading compass, a text field showing the active group behavior, speed and turn rate sliders (Horstmann, 2000) and most significantly, the Squad? checkbox which allows the user to set the entity to a mounted or unmounted state. When in the squad node, movements are sent relative to the aggregation entity. When not in the squad mode, movements are sent in independent real-world exercise coordinates. Corresponding entity trace text in the 3D environment changes color to match. Either way, human motion is smooth and continuous.

The last panel on the bottom right is the aggregated helicopter control panel shown in Figure 53. It contains the controls to simulate physically based helicopter flight as well as the features for aggregation discussed in the human panel above. The action interpreters support the mounting algorithm discussed in Chapter VI and also the rule-based behaviors with their associated cross-talk ability discussed in this chapter.

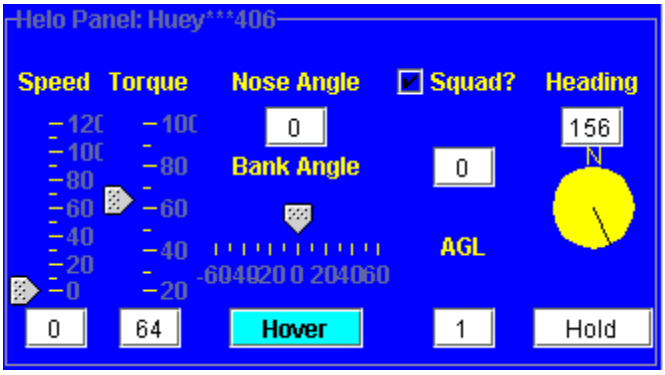


Figure 53. The Helicopter Panel subcomponent of the team control panel.

D. SUMMARY

This chapter discussed the implementation of the multiple-entity single-user interface. The use of rule based movements to define group behaviors in a networked VE was described. Individual behavior implementations to simulate tactical activities were described and illustrated. The master control panel, its components and the underlying action interpreters were also examined in this chapter.

VIII. CONCLUSIONS AND RECOMMENDATIONS

A. GENERAL THESIS CONCLUSIONS

The construction of a networked 3D visualization and planning tool with articulated virtual humans performing realistic individual and group behaviors is feasible. Content to support and extend such a tool can be rapidly generated and integrated, with behaviors being extensible and interchangeable. The tool's source code can be platform-independent, world-wide deployable and enjoys open-source distribution. As shown in Figure 54, this thesis contributes to and extends the 16-year body of work, in networked VE technology useful to DoD and civilian applications, performed by the NPSNET Research Group.

B. SPECIFIC CONCLUSIONS AND RESULTS

1. Inclusion of Articulated Humans

The H-Anim 1.1 spec exemplar Nancy.wrl, created by Cindy Ballreich, was used as the base model for this thesis project. By wrapping the EXTERNPROTO declaration of the EspduTransform node around the H-Anim PROTOs and content of the Nancy.wrl file and modifying the behavior library implementation through a Script node, a DIS-compliant human avatar was created. This composition of EXTERNPROTOs is an excellent, efficient and general result. A GUI interface for a single virtual human entity, single-user control and an action interpreter were implemented to add the DIS-compliant Nancy to the DIS-Java-VRML framework and instantiate the entity within CTF.

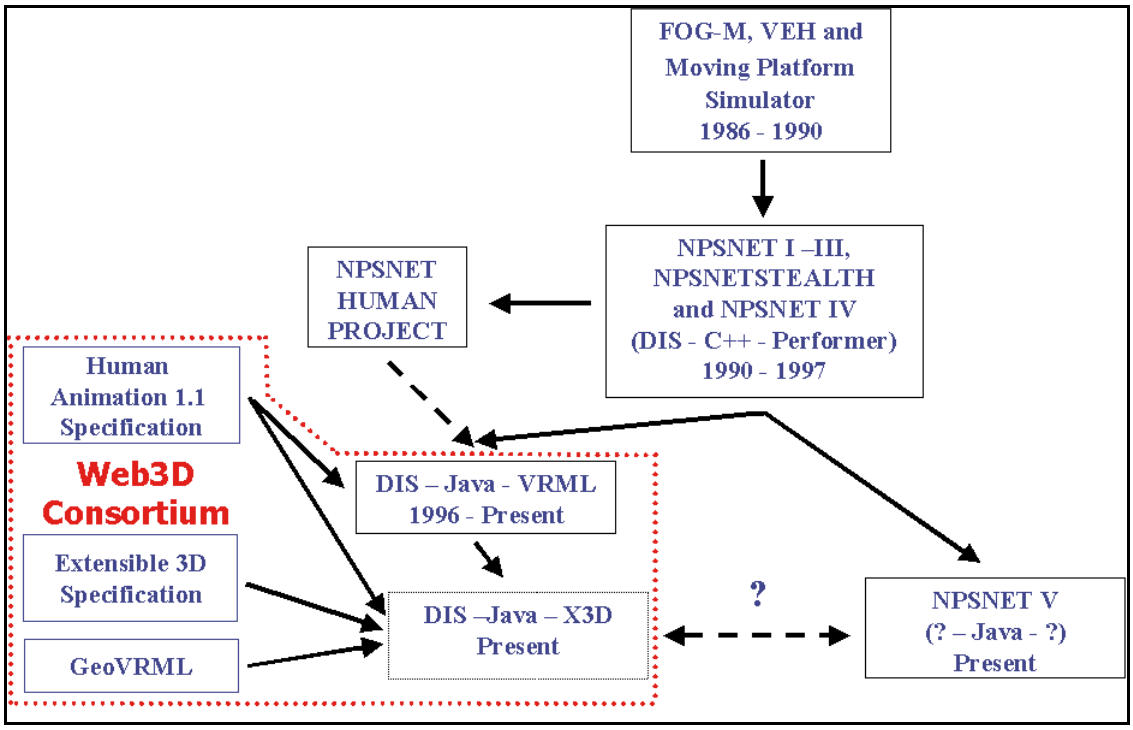


Figure 54. The Flow of NPSNET Research Group Systems showing this thesis research as part of the continuum.

2. Aggregation and Disaggregation

A basic mounting algorithm is now defined and implemented, utilizing a non-physical aggregation entity to control aggregated entities. The mounting algorithm is included for use in the action interpreters of the aggregation entity and the human and helicopter aggregating entities. A VRML scene graph concept for rendering the aggregated entities using either relative or world coordinates has been researched and implemented through the use of sophisticated VRML ROUTE addition and deletion under the control of a resident Script node.

3. Group Behaviors

Using the ability to aggregate and disaggregate, a series of group behaviors simulating tactical activities is now developed and implemented. This includes the development of a multi-entity, single user interface consisting of a GUI control panel and its subcomponent Java jPanels, and the implementation of action interpreters for the aggregating humans and helicopters and for the aggregation entity. The behaviors are implemented through a set of rules using goal positions and orientations and match two categories, single and compound. The single behaviors are the movement formations, Line, Wedge, and Column, and the control behaviors, None and Hold. The more complex compound behaviors use one or more of the single behaviors to define its actions. The compound behaviors consist of the air assault behaviors Mount Vehicle and Unmount Vehicle to the Right or Left, as well as the advanced movement formation, Bound.

4. Extensibility and Rapid Content Creation

The rapid creation and extensibility of content to include in the networked VE is illustrated through the use of the X3D specification and its associated authoring tool X3D-Edit. A native tag set for H-Anim 1.1 specification was created through the definition of the H-Anim PROTOs as EXTERNPROTOs and the development of an H-Anim 1.1 spec DTD. Forward compatibility from X3D to the H-Anim 1.1 spec and X3D's H-Anim compliance is demonstrated through both the PROTO declaration implementation and the implementation using the H-Anim native XML tag set. The extensibility and interchangeability of geometry and behaviors is demonstrated through

the development and addition of a new individual behavior, Kneel, to both the original and DIS-compliant virtual human avatar, Nancy.

5. Platform-Independent, Open-Source Distribution and World-Wide Deployable Source Code

The source code developed in this thesis project currently runs in VRML browsers on the Windows family of operating systems (Windows 95, Windows 98 version 2, NT 4.0, and Windows 2000) with reasonable performance on all platforms. The source code itself is available on the Web3d Consortium web site in its entirety in the DIS-Java-VRML working group download page (http://www.web3d.org/ WorkingGroups/vrtp/dis-java-vrml/ download.html) while the H-Anim work in XML can be found in the X3D task group examples page (http://www.web3d.org/ TaskGroups/x3d/translation/ examples.html). The source code is also included in the Appendix on CD-ROM and will be posted with an electronic copy of this thesis document on the NPSNET Research group website under the category of the MOVES Academic group Theses (http://www.npsnet.org/~moves/Theses.html).

C. LESSONS LEARNED

Legacy and archival code can be difficult to understand, work with and extend. The researcher's task of working with legacy code is greatly facilitated by clearly written, well-documented source code. In turn, as the researcher extends the original code, it is important that new changes are documented as part of a coherent body of work.

As a Masters student doing research in LSVEs, one is (generally) constrained to implement work within existing systems. The time to learn the system is greatly reduced by good documentation, which subsequently allows the student a longer period of time to conduct research. This thesis involved two significant pieces of legacy code, Cindy Ballreich's Nancy.wrl and the DIS-Java-VRML Working Group's code. In both cases, good documentation was provided because the code was designed with the goal of providing an examples for other authors to use. As students from the MOVES Academic Group and/or the NPSNET Research Group continue research in VE technologies, we must look upon good documentation as a mission critical task with the view and goal that our research should provide an example for other researchers. All Java code is required to have good Javadoc additions (Brutzman, 2000) to support clarity and reusability.

D. RECOMMENDATIONS FOR FUTURE WORK

1. Adaptive Agent

The addition of adaptive software agents can become a powerful extension to this thesis project. Currently the source code of the aggregated humans is designed in such a way to permit rapid conversion to run as software agents using the RELATE agent architecture developed by LCDR Kimberly Roddy, USN and LT Michael Dickson, USN (Roddy, 2000). The 3D implementation of networked virtual humans as software agents opens a large area of research in complex adaptive systems (Dörner, 1996) to include discovery of emergent behaviors in examining tactics, techniques and procedures, as well as the opportunity to do analytical and validation work on the efficacy of the agent

software simulation in comparison to current standard operating procedures. Two areas of specific future research involving agents are addressed below.

a) React to Environmental Stimuli

Ideally, even under real human control while conducting a combat simulation, these virtual humans need to react to environmental stimuli autonomously. By using an adaptive-agent implementation, the virtual humans might react autonomously to input stimuli such as incoming direct fire or indirect fire. The agents actions might be programmed to follow task and standards found for these actions in US Army or Marine Corps training and evaluation manuals.

b) React to Higher Level Behavior Commands (Auditory, Message and Visual Stimuli)

In addition to environmental stimuli, an agent implementation also needs to respond to commands provided from a higher command element or actions caused by higher-level behaviors. Examples include reaction to voice commands using a natural language interface, simulated radio messages received within the virtual world, or hand and arm signals given by a group commander. The autonomous response of agents to these conditions also provides a rich area of future research.

2. Increase Behavior and Motion Libraries

The H-Anim specification-compliant library of individual motions for the virtual humans needs to be increased to provide a greater range of motions and behaviors. Recommended immediate additions are prone and crouch positions, crawl, and walk/run while crouched motions. Later additions need to include more upper-body articulated

motions to conduct activities such as transferring equipment, as well as the ability to perform standard military hand and arm signals.

Future work to extend the group behaviors is essentially limitless, particularly if development focuses outside the specific tactical constraints addressed in this thesis. Higher levels of aggregation into larger units will require the development of more sophisticated compound behaviors. In support of US SOF and US Marine Corps, behaviors to allow small boat operations and amphibious operations need to be developed. Additionally, features addressed in the NPSNET-Human project, such as medical evacuation by ground and air, also provide a rich area for future research and development in cooperative and group behaviors.

3. Develop Geometry Library of Bodies, Vehicles and Attachments

An H-Anim specification-compliant geometry library containing a variety of human avatar and vehicle geometries needs to be developed to represent friendly and opposing forces. Additionally, other libraries of attachments to model and represent personal and unit military equipment (such as arms, ammunition, radios and explosives) need to be developed so that mission rehearsals and planning can be further enhanced. For longer-term development, these items can be made interactive so that equipment can be exchanged and cross loaded as well as operated within the parameters of a mission. An example of such an item for development is a laser target designator to guide the delivery of precision munitions against a target.

4. Rapid Creation World-Wide of Georeferenced Terrain

For a viable networked 3D planning tool, the area of operations where a mission will be conducted must be capable of rapid computer generation. The GeoVrml specification and its inclusion in the X3D standard provides a proven approach to further this concept through future research. Work currently under way at NPS involves a terrain server containing a digitized terrain map for the entire world land mass. Such work is an essential step toward the rapid development of geolocated virtual worlds for operations planning.

5. Development of Drag and Drop Object Library for Rapid Scene or World Development and Modification

In pursuing the goal of a networked 3D planning tool, the ability to rapidly add world objects is essential to mission planning. The Cortona VRML Client currently supports drag and drop technology. The addition of this feature into the CTF framework in concert with the development of a library to include man-made objects, like roads and buildings, and natural objects, such as vegetation, greatly enhances its potential as a planning tool.

6. Inclusion of Line of Sight and Detection Algorithms

In support of a mission-planning tool, it is essential to include real world line of sight and detection algorithms. The development and inclusion of these capabilities especially when involving georeferenced terrain provides a powerful and necessary enhancement to worlds developed in the DIS-Java-VRML framework.

7. Scale-Up Level of Aggregation

The development of larger units and their associated behaviors is another important area of future work. This thesis developed a proof of concept using a fire team level of organization. Network scalability issues, command and control issues, and level-of-detail work to speed rendering of a many entity scene requires further research and development. The Q-U-I-C-K algorithm is a particularly promising avenue to continue (Capps, 2000).

8. Predictive Positional PDU

As a method to arbitrate network-latency problems involved with rapid orientation and heading changes of the aggregation entity when combined with actions conducted in close proximity to the aggregation entity, a predictive positional PDU needs to be developed. Such a PDU might be issued by the aggregation entity to inform the aggregated entity of an anticipated upcoming (or non-linear) state change. Such a PDU might be particularly powerful when humans implemented as agents can react to environmental stimuli. For example, as the unit approaches an obstacle, the aggregation entity can inform the aggregated entities of an expected direction change or halt. Greater synchronicity among distributed participants will also occur.

9. Mounting Algorithm Used for Deployment Load Planning for Ships and Airplanes

This research on mounting and aggregation can be extended to other fields where 3D visualization is useful. An example implementation is the creation of a tool used for deployment load planning and execution, or amphibious operations. Such a tool might

allow the user to develop a load/unload plan based off the available means of transportation. The deployment items could be loaded into the virtual transport ensuring that all items are transportable in regards to volume and weight. Additionally, weight and balance calculations might be conducted to maximize and optimize the transports' ability to move the load. At the unload site, users might have a the ability to identify locations of specific items by clicking a manifest list and having the items highlight in the 3D visualization. Additionally, optimized unloading plans, again taking into account the transports' weight and balance parameters, might then be developed for either most rapid outload or based off the requirement for designated critical equipment to be outloaded first.

10. Autogenerated LSVEs For Inclusion in Operations Orders

Much of the previously discussed future work is a subcomponent of the final, big-picture production to rapidly autogenerate networked 3D LSVEs for inclusion as annexes to military operations orders operations plans or contingency plans. By developing and implementing these subcomponents, we are incrementally advancing the possibility of such a dramatic product in the near-term future. Such capabilities and achievements are the penultimate goal of this research.

APPENDIX A. CD-ROM

Theses Appendix Published
as Part of the
Distributed Interactive Simulation
DIS-Java-VRML Working Group

X3D and DIS-Java-VRML Theses

MAJ David W. Laflam
US Army

Title:3D Visualization of Theater-Level Radio Communications Using a Networked Virtual Environment

ABSTRACT

The military is heavily reliant on the transfer of information among various networks in day-to-day operations. Radio-based communications networks that support this volume of information are complex, difficult to manage, and change frequently. Communications network planners need a way to clearly visualize and communicate mobile operational network capabilities, particularly to network users.

By using the DIS-Java-VRML simulation and modeling toolkit, visualizations of radio-frequency energy and radio path-profiling data can be quickly generated as 3D models. These animated 3D visualizations can be loaded into a networked virtual environment, so that communications planners can detect a variety of problems such as radio frequency interference and gaps in coverage. Planners can also brief senior staff, plan within their own staff, and collaborate with communications staff planners in distant locations using such virtual environments.

DIS-Java-VRML visualization tools can provide a clear picture of the battle space with respect to the deployed communications architecture. The prototypes presented in this thesis demonstrate the ability to generate

a shared visualization that can show a radio communications network in 3D. Such dynamic visualizations increase communications planning information bandwidth and yield more intuitive ways of presenting information to users. Higher information density in a more intuitive format enables better understanding with quicker reaction times. This thesis and the visualization tool discussed provide the foundation for fundamental improvements in visualizing radio communications environments.

Thesis Link
Software Reference Link

MAJ Thomas E. Miller
US Army

Integrating Realistic Human Group Behaviors Into A Networked 3d Virtual Environment

ABSTRACT

Virtual humans operating inside large-scale virtual environments (VE) are typically controlled as single entities. Coordination of group activity and movement is usually the responsibility of their "real world" human controllers. Georeferencing coordinate systems, single-precision versus double-precision number representation and network delay requirements make group operations difficult. Mounting multiple humans inside shared or single vehicles, (i.e. air-assault operations, mechanized infantry operations, or small boat/riverine operations) with high fidelity is often impossible.

The approach taken in this thesis is to reengineer the DIS-Java-VRML Capture the Flag game geolocated at Fort Irwin, California to allow the inclusion of human entities. Human operators are given the capability of aggregating or mounting nonhuman entities for coordinated actions. Additionally, rapid content creation of human entities is addressed through the development of a native tag set for the Humanoid Animation (H-Anim) 1.1 Specification in Extensible 3D (X3D). Conventions are demonstrated for integrating the DIS-Java-VRML and H-Anim draft standards using either VRML97 or X3D encodings.

The result of this work is an interface to aggregate and control articulated humans using an existing model with a standardized motion library in a networked virtual environment. Virtual human avatars can be mounted and unmounted from aggregation entities. Simple demonstration examples

show coordinated tactical maneuver among multiple humans with and without vehicles. Live 3D visualization of animated humanoids on realistic terrain is then portrayed inside freely available web browsers.

Thesis Link
Software Reference Link

MAJ Mark Murray
MAJ Jason Quigley
US Air Force

Automatically Generating A Distributed 3d Battlespace Using Usmtf And Xml-Mtf Air Tasking Order, Extensible Markup Language (Xml) And Virtual Reality Modeling Language (Vrml)

ABSTRACT

For the past three decades, the Department of Defense (DoD) has used the U.S. Message Text Format (USMTF) as the primary means to exchange information and to achieve interoperability between joint and coalition forces. To more effectively exchange and share data, the Defense Information Systems Agency (DISA), the lead agency for the USMTF, is actively engaged in extending the USMTF standard with a new data sharing technology called Extensible Markup Language (XML). This work translates and synthesizes Air Tasking Order (ATO) data messages written in XML into a three-dimensional (3D) air attack plan within a virtual environment through the use of the Virtual Reality Modeling Language (VRML).

Thesis Link
Software Reference Link

X3D and DIS-Java-VRML Software Tools

A. **X3DEdit**

B. **X3DEdit Examples**

C.**JDK.1.2.2**

D. **Netscape 4.73**

E. **COSMO Player**(VRML Plug-in)

F. **KELP Forest**

G. **VRTP** Distribution

H. **RRA**(Recursive Ray Acoustic)

I. **Xeena 1.2**

J. **Vorlon**(VRML Syntax Checker)

K. **Cygwin** (Windows Unix Tools)

L. **DIS-Java-VRML**

M. **Xj3D**

N. **GeoVRML**

A. X3DEdit

X3D-Edit is a graphics file editor for Extensible 3D (X3D) that enables simple error-free editing, authoring and validation of X3D or VRML scene-graph files.

Local URL: **www.web3D.org/TaskGroups/x3d/translation/README.X3D-Edit.html**

Local Fie Download:**X3D-Edit.zip**

URL: **http://www.web3d.org/x3d.html**

B. X3DEdit Examples

Local Fie Download:**X3D-Examples.zip**

These are the examples for X3D-Edit based on VRML 2.0 Sourcebook by David Nadue

---

C. JDK.1.2.2

The essential Java 2 SDK, tools, runtimes, and APIs for developers writing, deploying, and running applets and applications in the Java programming language. Also includes earlier Java Development Kit versions JDKTM 1.1 and JRE 1.1

Local Fie Download: **j2sdk1_3_0-win.exe**

URL: **http://java.sun.com/products/index.html**

D. Netscape 4.73

Communicator 4.7 is the latest release of the Internet software suite from Netscape. In addition to the Netscape Navigator browser, Communicator includes a complete set of tools for effective everyday communication.

Local Fie Download: **netscape-cc32d475.exe**

URL: **http://home.netscape.com/computing/download/index.html**

E. COSMO Player (VRML Plug-in)

Cosmo Player is a high-performance, cross-platform VRML 2.0 client designed for fast and efficient viewing of virtual worlds. Navigate and manipulate 3D scenes and bring your Web experience to a new level.

Use VRML to fly through anatomy class, experience 3D data visualizations, or show off a CAD model. Cosmo Player is the premiere viewing client for VRML, with support for the latest standards.

Whether on the Internet or in an enterprise, Cosmo Player allows web content creators and applications developers to add visual and multimedia elements to their work.

Local Fie Download: **cosmo_win95nt_eng.exe**

URL: **http://www.cai.com/cosmo/html/win95nt.htm**

F. KELP Forest

Two classes of graduate students learning 3D graphics and analytic simulation at the Naval Postgraduate School modeled the three-dimensional (3D) shape, structure, imagery and motion behaviors of plants and animals in the Kelp Forest Exhibit at the Monterey Bay Aquarium. Our intended audience includes educators and students of all ages, scientific users interested in composing models in a 3D Web environment, and the general public. By focusing on thoroughly modeling a controlled environment, we produced an exemplar 3D graphics site for modeling larger and more sophisticated underwater domains. The Virtual Reality Modeling Language (VRML) proved to be an excellent medium for capturing diverse models, composing multiple student efforts, and

publishing dynamic results publicly on the Web. This project was successfully demonstrated to 1000 people during the National Ocean Fair in Monterey June 12 1998.

Local URL: **kelp/index.html**

Local Fie Download: **kelp.zip**

URL: **http://web.nps.navy.mil/~brutzman/kelp**

G. VRTP Distribution

The capabilities of the Virtual Reality Modeling Language (VRML) permit building large-scale virtual environments using the Internet and the World Wide Web. However the underlying network support provided by the hypertext transfer protocol (http) is insufficient for large-scale virtual environments. Additional capabilities for many-to-many peer-to-peer communications plus network monitoring need to be combined with the client-server capabilities of http. To accomplish this task, we present a detailed design rationale for the virtual reality transfer protocol (vrtp). vrtp is designed to support interlinked VRML worlds in the same manner as http was designed to support interlinked HTML pages. vrtp will be optimized in two ways: on individual desktops and across the Internet. vrtp appears to be a necessary next step in the deployment of all-encompassing interactive internetworked 3D worlds.

Local URL: **vrtp/vrtp/index.html**

Local Fie Download: **vrtp.zip**

URL: **http://www.web3d.org/WorkingGroups/vrtp/**

H. RRA (Recursive Ray Acoustic)

This project calculates and renders physically realistic sonar beams in real time. Java programs are used for sonar ray-tracing computation and the Virtual Reality Modeling Language (VRML 97) is used for 3D graphics.

The primary motivation for this project is to produce underwater sonar beams for analytic and visualization use in virtual worlds. Virtual world simulations are realistic when individual components are simulated in a manner that reflects reality. For an underwater virtual world that includes simulated acoustic detection, a physically based sonar propagation model is required if ranges in excess of tens of meters are expected. The

Recursive Ray Acoustics (RRA) Algorithm by Dr. Lawrence Ziomek of NPS provides a general & rapid ray-tracing algorithm which can accurately & quickly predict sonar propagation through seawater, under a wide variety of surface, water-column and ocean-bottom environmental conditions.

This project creates an application programming interface (API) for real-time 3D computation and visualization of acoustic energy propagation. The API provides features for generating complex physically based sonar information at interaction rates, and then visualizing that acoustic information. The simulation is programmed in Java, and runs either as a stand-alone program or as a script in a web browser. This program generates Virtual Reality Modeling Language (VRML 97) compliant code that can be viewed from any VRML-capable Web browser. This approach allows the characteristics of the energy propagation to be calculated with high precision and observed in three dimensions (3D) and in real time.

As sonar-system information bandwidth becomes larger, more intuitive ways of presenting information to users are required. Interactive 3D graphics with environmental and entity rendering can free users from from mentally integrating complex data piecemeal. This approach can enable significantly greater understanding and quicker reaction times. We are optimistic that this API might someday provide the foundation for fundamental advances in sonar modeling and visualization.

Local URL: **rra/vrtp/rra/rra.html**

Local Fie Download: **rra.zip**

URL: **http://web.nps.navy.mil/~brutzman/vrtp/rra/rra.html**

I. Xeena 1.2

Xeena is a generic Java application from the IBM Haifa Research Laboratory for editing valid XML documents derived from any valid DTD. The editor takes as input a given DTD, and automatically builds a palette containing the elements defined in the DTD. Users can thus create/edit/expand any document derived from that DTD, by using a visual tree-directed paradigm. The visual paradigm requires a minimum learning curve as only valid constructs/elements are presented to the user in a context-sensitive palette. A Key feature of Xeena is its syntax directed editing ability. Xeena is aware of the DTD grammar, and by making only authorized elements icons sensitive, automatically ensures that all documents generated are valid according to the given DTD.

Local Fie Download: **Xeena-1.2EA.exe**

URL: **http://www.alphaworks.ibm.com/tech/xeena**

J. Vorlon(VRML Syntax Checker)

The industry standard command line VRML Validator
Trapezium developed Vorlon as a service to the VRML community to allow authors to spend less time chasing bugs and more time creating high quality content.
Validates conformance to VRML97 specification
Displays line, line number and descriptive message
for each error or warning
Vorlon is Freeware.

Local Fie Download: **vorlon.exe**

URL: **http://www.trapezium.com/VorlonPage.htm**

K. Cygwin (Windows Unix Tools)

Cygwin brings a standard UNIX/Linux shell environment, including many of its most useful commands, to the Windows platform so IT managers can effectively deploy trained staff, and leverage existing investments in UNIX/Linux source code and shell scripts.

URL: **http://www.cygnus.com/cygwin/**

L. DIS-Java-VRML

The area of interest of this working group is the nexus of DIS, Java and VRML. The IEEE Distributed Interactive Simulation (DIS) Protocol is used to communicate state information (such as position, orientation, velocities and accelerations) among multiple entities participating in a shared network environment. Java is a portable networked programming language that can interoperate on any computer which includes a Web browser. The Virtual Reality Modeling Language (VRML) enables platform-independent interactive three-dimensional (3D) graphics across the Internet, and can be used to compose sophisticated 3D virtual environments.
The DIS-Java-VRML Working Group is developing a free software library, written in Java and interoperable with both DIS and VRML. There are a number of people contributing to the public-domain code archive. This software is protected under the terms of the GNU General Public License.

Local URL: **vrtp/dis-java-vrml/index.html**

Local Fie Download:**dis-java-vrml.zip**

URL: **http://www.web3d.org/WorkingGroups/vrtp/dis-java-vrml/**

M. Xj3D

Xj3D is an example implementation for the X3D specification. Specifically, Xj3D is a Java3D-based open-source loader, browser and exporter for Extensible 3D (X3D) graphics.

URL: **http://web3d.metrolink.com/cgi-bin/cvsweb.cgi/x3d/HowToInstall.html**

N. GeoVRML

GeoVRML is an official Working Group of the Web3D Consortium. It was formed on 27 Feb 1998 with the goal of developing tools and recommended practice for the representation of geographical data using the Virtual Reality Modeling Language (VRML). The desire is to enable geo-referenced data, such as maps and 3-D terrain models, to be viewed over the web by a user with a standard VRML plugin for their web browser.

Local URL: **GeoVRML/1.0/doc/index.html**

Local Fie Download: **geovrml1_0.exe**

URL: **http://www.ai.sri.com/geovrml/**

September 24 2000 (**official NPS disclaimer**)
URL: **www.web3D.org/WorkingGroups/vrtp/dis-java-vrml/SoftwareReference.html**
feedback: **feedback@nps.navy.mil**

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Abbot, E.A., *Flatland: A Romance of Many Dimensions,* Dover Publications, Inc., 1992, (orig. 1884).

Adler, S., and others, "Extensible Stylesheet Language (XSL), Version 1.0, W3C Working Draft." [http://www.w3.org/TR/xsl/]. 27 March 2000.

Ames, A.L., Nadeu, D.R., and Moreland, J.L., *VRML2.0 Sourcebook,* John Wiley & Sons, Inc, 1997.

Badler, N.I., Phillips, C.B. and Webber, B.L., *Simulating Humans: Computer Graphics Animation and Control,* Oxford University Press, 1993.

Bos, B., "XML in 10 Points." [http://www.w3.org/XML/1999/XML-in-10-points]. 26 May 2000.

Brutzman, D., "The Virtual Reality Modeling Language and Java," *Communications of the ACM,* v. 41(6), pp. 57-64, June 1998.

Brutzman, D. "Internetworked Graphics: Capabilities, Shortfalls, Frontiers," *Course Notes from SIGGRAPH 2000, Course 28: Internetworked 3D Computer Graphics: Overcoming Bottlenecks and Supporting Collaboration,* New Orleans, LA, 23-28 July 2000.

Capps, M. V., *Fidelity Optimization in Distributed Virtual Environments,* Ph.D. Dissertation, Naval Postgraduate School, Monterey, California, June 2000.

Chrislip, A.C. and Ehlert, J.F., Jr., *Level of Detail Models for Dismounted Infantry in NPSNET-IV.8.1,* Master's Thesis, Naval Postgraduate School, Monterey, California, September 1995.

Electronic mail correspondences between David Chamberlin, Don Brutzman, others, and the author, 29-30 April 2000. A hypermail archive of this correspondence at [http://www.web3d.org/workingGroups/vrtp/dis-java-vrml/hypermail/2000].

Dörner, D., *The Logic of Failure: Recognizing and Avoiding Error in Complex Situations,* Addison-Wesley, 1996.

Extensible 3D (X3D) Task Group, [http://www.web3d.org/fs_workinggroups.htm]. August 2000.

Halvorson, J.A., *Realistic Interface and Control of a Virtual Submarine in NPSNET,* Master's Thesis, Naval Postgraduate School, Monterey, California, March 1997.

Horstmann, C.S. and Cornell, G., *Core Java 2: Volume I-Fundamentals,* Sun Microsystems Press (Java Series), 1999.

Horstmann, C.S. and Cornell, G., *Core Java 2: Volume II-Advanced Features,* Sun Microsystems Press (Java Series), 2000.

Human Animation Working Group, "Specification for a Standard Humanoid, Version 1.1.", [http://ece.uwaterloo.ca/~h-anim/spec1.1/]. 3 August 1999.

IBM, *Alpha Works,* [http://www.alphaWorks.ibm.com/tech/xeena], September 2000.

IEEE Standard for Information Technology – Protocols for Distributed Interactive Simulation (DIS) Applications, v. 2, Institute for Simulation and Training Report IST-CR-93-15, University of Central Florida, Orlando, Florida, 28 May 1993,

IEEE Standard for Distributed Interactive Simulation – Application Protocols, IEEE Standard 1278.1-1995, IEEE, Inc., 1995.

Macedonia, M.R., Zyda, M.J., Pratt, D.R., Barham, P.T. and Zeswitz, S., "NPSNET: A Network Software Architecture for Large Scale Virtual Environments," *PRESENCE: Teleoperators and Virtual Environments,* v. 3(4), pp. 265-287, Fall 1994.

Macedonia, M.R., Brutzman, D.P, Zyda, M.J., Pratt, D.R., Barham, P.T., Falby, J. and Locke J., "NPSNET: A Multi-Player 3D Virtual Environment Over The Internet," *Proceedings of the 1995 Symposium on Interactive 3D Graphics*, April 1995.

Maestri, G., *[digital] Character Animation 2: Volume 1 – essential techniques,* New Riders Publishing, 1999.

Marine Corps Doctrinal Publication (MDCP) 1, *Warfighting*, 1997.

McCloud, S., *Understanding Comics: The Invisible Art,* Kitchen Sink Press, 1993.

McCloud, S. *Reinventing Comics: How Imagination and Technology Are Revolutionizing an Art Form,* HarperCollins Books, 2000.

McGregor, D., "DIS-Java-VRML: A Relatively Painless Introduction," *Tutorial Slide Set from Web3D/VRML 2000, Monterey, California,* 52 slides, 21-24 February 2000.

Multigen-Paradigm, Inc., *Products,* [http://www.multigen.com/products/prodindex.htm], September 2000.

NPSNET Research Group, *NPSNET IV Software Download, NPSNETData Files:Naval Postgraduate School Campus, [http://www.npsnet.nps.navy.mil/npsnet0/ distribution/data/npsnetIV.10.3.NPS_Campus.data.tar.Z*], September 2000.

O'Byrne, J.E., *Human Interaction Within a Virtual Environment for Shipboard Training,* Master's Thesis, Naval Postgraduate School, Monterey, California, September 1995.

Parallel Graphics, *Products,* [http://www.parallelgraphics.com/products/], September 2000.

Pew, R.W. and Mavor, A.S., eds., *Modeling Human and Organizational Behavior: Applications to Military Simulations,* National Academy Press, 1998.

Pratt, D.R., Barham, P.T., Locke, J., Zyda, M.J. and others, "Insertion of an Articulated Human into a Networked Virtual Environment," *Proceedings of the 1994 Artificial Intelligence, Simulation and Planning in High Autonomy Systems Conference,* 7-9 December 1994.

Rand Note N-3536-SOCOM/JS, *Analysis of Special Operations Forces in Decision Aids: Current Shortfalls,* by B. Pirnie and M.C. Harrell, Approved for Public Release; distribution unlimited, 1994.

Reece, D.A. and Dumanoir, P., "Conventions for Representing Humans in a DIS Exercise: The DWN Experience." [http://www.asset.com/orl/disaf/papers/98f-siw-245_toc.htm]. July 2000.

Roddy, K. and Dickson, M., *Modeling Human And Organizational Behavior Using A Relation-Centric Multi-Agent System Design Paradigm,* Master's Thesis, Naval Postgraduate School, Monterey, California, September 2000.
Also available at:
[http://www.npsnet.org/~moves/Theses/RoddyDickson.pdf].

Roehl, B., "Distributed Virtual Reality – An Overview," *Proceedings of VRML '95,* pp. 39-43, 1995.

Roehl, B., "VRML Humanoids: A set of conventions for the creation of humanoid characters," *VRML Developer's Journal,* v. 1(1), pp. 8 –10, 12, & 14, 2 March 2000.
Also available at:
[http://www.sys-con.com/vrml/index.html]. March 2000.

Shockley, J.W., and Morganthaler, M., "Using H-ANIM to Represent Human Figures in HLA-Compliant Simulations," *Proceedings of the Spring 2000 Simulation Interoperability Workshop,* paper 00S-SIW-061, 26-31 March 2000.

Singhal, S. and Zyda, M., *Networked Virtual Environments: Design and Implementation*, ACM Press (SIGGRAPH Series), 1999.

Stewart, B.C., *Mounting Human Entities to Control and Interact with Networked Ship Entities in a Virtual Environment,* Master's Thesis, Naval Postgraduate School, Monterey, California, March 1996.

Transom Corporation, "Image Gallery," [http://www.transom.com/Public/picturegallery.shtml]. August 2000.

U.S. Army Field Manual (FM) 34-130, *Intelligence Preparation of the Battlefield,* 1994.

Waldrop, M.S., *Real-Time Articulation of the Upper Body for Simulated Humans in Virtual Environments,* Master's Thesis, Naval Postgraduate School, Monterey, CA, September 1995.

ZdNet, *GameSpot 3D Model Gallery,* [http://www.gamespot.com/3dgallery/tribes-medium-female.wrl]. September 2000.

Zyda, M.J., Pratt, D.R., Pratt, S., Barham, P. and Falby, J.S., "NPSNET-HUMAN: Inserting The Human Into The Networked Synthetic Environment," *Proceedings of the 13th DIS Workshop,* pp. 103-106, 18-22 September 1995.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center……...…….…….…..…2
   8725 John J. Kingman Road, STE 0944
   Fort Belvoir, VA 22060-6218

2. Dudley Knox Library..……...…….2
   Naval Postgraduate School
   411 Dyer Road
   Monterey, CA 93943-5101

3. Dr. Don Brutzman, Code UW/Br..……...……..…….1
   Naval Postgraduate School
   Monterey, CA 93943-5101

4. Dr. Michael J. Zyda, Code CS/Zk..……..……..1
   Chair, Modeling, Virtual Environments and Simulation (MOVES)
   Naval Postgraduate School
   Monterey, CA 93943-5101

5. LT Patrick Mack……1
   310 Aachen Road
   Seaside, CA 93955

6. COL Joseph Andrade, USA..……1
   Naval Postgraduate School
   Monterey, CA 93943-5101

7. Dr. Michael P. Bailey……1
   Technical Director, Marine Corps Training and Education Command
   Commanding General
   Marine Corps Combat Development Command, Code 46T
   3300 Russell Road
   Quantico, VA 22134

8. Cindy Ballreich ……..……...1
   1191 Rutland Road
   Newport Beach, CA 92660

9. Dr. Philip S. Barry……..…1
   Chief, S&T Initiatives Division
   Defense Modeling and Simulations Office
   1901 N. Beauregard Street, Suite 500
   Alexandria, VA 22311

10. Robert J Barton III……1
    Fraunhofer Center for Research in Computer Graphics (CRCG)
    321 South Main Street
    Providence, RI 02903

11. Curtis Blais……....1
    Institute for Joint Warfare Analysis
    Naval Postgraduate School
    Monterey, CA 93943-5101

12. Rex Buddenberg, Code Is/Bu……..……..1
    Naval Postgraduate School
    Monterey, CA 93943-5101

13. CAPT Steve Chapman, USN……1
    N6M
    2000 Navy Pentagon
    Room 4C445
    Washington, DC 20350-2000

14. Erik Chaum……1
    NAVSEA Undersea Warfare Center
    Division Newport
    Code 2231, Building 1171-3
    1176 Howell Street
    Newport, RI 02841-1708

15. COL Steven Collier, USA……1
    Army Model and Simulation Office
    1111 Jefferson Davis Highway. Suite 503E
    Arlington, VA 22202

16. Dr. James Eagle……..……1
    Chair, Undersea Warfare Academic Group
    Naval Postgraduate School
    Monterey, CA 93943-5101

17. Dr. Paul Fishwick……..……1
Computer & Information Science and Engineering Department
University of Florida
Post Office Box 116120
322 Building CSE
Gainesville, FL 32611-6120

18. Connell Gallagher……..……1
President
ParallelGraphics
36 Upper Fitzwilliam Street
Dublin 2, Ireland

19. CDR Arthur Galpin, USN……..……1
US Special Operations Command (SORR-SCS)
7701 Tampa Point Boulevard
MacDill Air Force Base, FL 33621-5323

20. Jerry Ham……1
National Simulation Center (NSC)
ATTN: ATZL-NSC
410 Kearny Avenue, Building 45
Fort Leavenworth, KS 66027-1306

21. Dr. Tony Healey, Code ME/Hy……..……..1
Naval Postgraduate School
Monterey, CA 93943-5101

22. John Hiles, CS/Hj……..……....1
Naval Postgraduate School
Monterey, CA 93942-5101

23. David Holland……..……1
Computer Science/C3I Center MS4 A5
George Mason University
Fairfax, VA 22032

24. Pamela Krause……..1
Advanced Systems & Technology
National Reconnaissance Office
14675 Lee Road
Chantilly, VA 20151-1714

25. John Lademan……1
Electronic Sensors and Systems Sector
Northrop Grumman Corporation
PO Box 1488 – MS 9030
Annapolis, MD 21404

26. Jaron Lanier……..1
Advanced Network & Services, Inc.
200 Business Park Drive
Armonk, NY 10504

27. Dr. R. Bowen Loftin……..1
Director of Simulation Programs
Virginia Modeling Analysis & Simulation Center
Old Dominion University
7000 College Drive
Suffolk, VA 23435

28. CAPT Arnold O. Lotring, USN……1
Commanding Officer
Naval Submarine School
Code 00, Naval Submarine School
Post Office Box 700
Groton, CT 06349-5700

29. Dr. Michael Macedonia……1
Chief Scientist and Technical Director
US Army STRICOM
12350 Research Parkway
Orlando, FL 32826-3276

30. Michael McCann……1
Monterey Bay Aquarium Research Institute (MBARI)
Post Office Box 628
Moss Landing, CA 95039-0628

31. CDR John C. Mickey, USN……..1
PEO for Submarines (PMS401)
2531 Jefferson Davis Highway
NC3, Room 3W30
Arlington, VA 22242-5161
ATTN: CDR John Mickey

32. Dr. Henry T. Miller ……..…1
935 Lakeview Drive
Paducah, KY 42003

33. Dr. Tyrus H. Miller……...……..…...1
504 Hagar Court
Santa Cruz, CA 95064

34. CAPT William Molloy, USN……1
Chief Modeling & Simulation
Joint Warfighting Center
US Joint Forces Command
116 Lake View Parkway
Suffolk, VA 23435-2697

35. CPT Mark Murray, USAF……1
Joint Battlespace Infosphere, (JBI)
AFRL/IFSE
Building 3, Room E-1078
525 Brooks Road
Rome, NY 13441-4505

36. Michael Myjak……..1
Vice President and CTO
The Virtual Workshop
Post Office Box 98
Titusville, FL 32781

37. LTC Joel Parker, USA……..……..1
Naval Postgraduate School
Monterey, CA 93943-5101

38. George Phillips……..1
CNO, N6M1
2000 Navy Pentagon
Room 4C445
Washington, DC 20350-2000

39. Dr. Mark Pullen……...1
Department of Computer Science/C3I Center MS4A5
George Mason University
Fairfax, VA 22030

40. CPT Jason Quigley, USAF……1
Joint Battlespace Infosphere, (JBI)
AFRL/IFSE
Building 3, Room E-1078
525 Brooks Road
Rome, NY 13441-4505

41. Dr. Martin Reddy……..1
SRI International, EK219
333 Ravenswood Avenue
San Jose, CA 94025

42. Bernie Roehl..……...……...……..…….……..1
68 Margaret Avenue North
Waterloo, Ontario
N2J 3P7
Canada

43. Dr. R. Jay Roland……..1
President, Roland and Associates
500 Sloat Avenue
Monterey, CA 93940

44. MAJ Glenn Roussos, USA...…….……1
US Special Operations Command (SORR-SCS)
7701 Tampa Point Boulevard
MacDill Air Force Base, FL 33621-5323

45. Dr. Sandeep Singhal……...…..1
ReefEdge, Inc.
96 Linwood Plaza, PMB 505
Fort Lee, NJ 07024-3701

46. Keith Victor ……..……1
Virtock Technologies, Inc
551 Belle Meade Farm Drive
Loveland, OH 45140

47. CAPT Robert Voigt, USN..……1
Chair, Electrical Engineering Department
US Naval Academy
Annapolis, MD 21402

48. Walter H. Zimmers……1
Defense Threat Reduction Agency
CPOC
6801 Telegraph Road
Alexandria, VA 22310-3398

49. MAJ Thomas E. Miller …….……..…...1
Post Office Box 3101
Fort Leavenworth, KS 66027